ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ

ΣΧΟΛΗ ΘΕΤΙΚΩΝ ΕΠΙΣΤΗΜΩΝ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ ΜΕ ΕΦΑΡΜΟΓΕΣ ΣΤΗ ΒΙΟΙΑΤΡΙΚΗ

Algorithm for Identifying Small Open Reading Frames in DNA

Tsapara Despoina

BACHELOR'S THESIS

Thesis Advisor

Hatzigeorgiou Artemis

Higher Education

Lamia, 2023

Με ατομική μου ευθύνη και γνωρίζοντας τις κυρώσεις [(1)], που προβλέπονται από της διατάξεις της παρ. 6 του άρθρου 22 του Ν. 1599/1986, δηλώνω ότι:

1. *Δεν παραθέτω κομμάτια βιβλίων ή άρθρων ή εργασιών άλλων αυτολεξεί* ***χωρίς να τα περικλείω σε εισαγωγικά*** *και χωρίς να αναφέρω το συγγραφέα, τη χρονολογία, τη σελίδα. Η αυτολεξεί παράθεση χωρίς εισαγωγικά χωρίς αναφορά στην πηγή, είναι λογοκλοπή. Πέραν της αυτολεξεί παράθεσης, λογοκλοπή θεωρείται και η παράφραση εδαφίων από έργα άλλων, συμπεριλαμβανομένων και έργων συμφοιτητών μου, καθώς και η παράθεση στοιχείων που άλλοι συνέλεξαν ή επεξεργάσθηκαν, χωρίς αναφορά στην πηγή. Αναφέρω πάντοτε με πληρότητα την πηγή κάτω από τον πίνακα ή σχέδιο, όπως στα παραθέματα.*

2. *Δέχομαι ότι η αυτολεξεί* ***παράθεση χωρίς εισαγωγικά****, ακόμα κι αν συνοδεύεται από αναφορά στην πηγή σε κάποιο άλλο σημείο του κειμένου ή στο τέλος του, είναι αντιγραφή. Η αναφορά στην πηγή στο τέλος π.χ. μιας παραγράφου ή μιας σελίδας, δεν δικαιολογεί συρραφή εδαφίων έργου άλλου συγγραφέα, έστω και παραφρασμένων, και παρουσίασή τους ως δική μου εργασία.*

3. *Δέχομαι ότι υπάρχει επίσης περιορισμός στο μέγεθος και στη συχνότητα των παραθεμάτων που μπορώ να εντάξω στην εργασία μου εντός εισαγωγικών. Κάθε μεγάλο παράθεμα (π.χ. σε πίνακα ή πλαίσιο, κλπ), προϋποθέτει ειδικές ρυθμίσεις, και όταν δημοσιεύεται προϋποθέτει την άδεια του συγγραφέα ή του εκδότη. Το ίδιο και οι πίνακες και τα σχέδια*

4. *Δέχομαι όλες τις συνέπειες σε περίπτωση λογοκλοπής ή αντιγραφής.*

Ημερομηνία: …15…/.2../ 2023

Ο – Η Δηλ.

(Υπογραφή)

(1) «Όποιος εν γνώσει του δηλώνει ψευδή γεγονότα ή αρνείται ή αποκρύπτει τα αληθινά με έγγραφη υπεύθυνη δήλωση του άρθρου 8 παρ. 4 Ν. 1599/1986 τιμωρείται με φυλάκιση τουλάχιστον τριών μηνών. Εάν ο υπαίτιος αυτών των πράξεων σκόπευε να προσπορίσει στον εαυτόν του ή σε άλλον περιουσιακό όφελος βλάπτοντας τρίτον ή σκόπευε να βλάψει άλλον, τιμωρείται με κάθειρξη μέχρι 10 ετών.

Algorithm for Identifying Small Open Reading Frames in DNA

Tsapara Despoina

Three-member Committee:

Bagos Pantelis, Higher Education

Braliou Georgia, Higher Education

Hatzigeorgiou Artemis, Higher Education (Thesis Advisor)

ACKNOWLEDGMENTS

I would like to extend my heartfelt gratitude to my advisor, Artemis Hatzigeorgiou, for providing me with the opportunity to undertake this assignment and for her invaluable guidance and inspiration throughout my academic journey.

I would also like to extend my gratitude to Professor Nikos Perdikopanis, who provided valuable insights and support throughout the entire process of this project. Their guidance and expertise were crucial in the successful completion of this project.

I would also like to extend my appreciation to my best friend, Eleonora Ieronimaki, for providing both precious help and encouragement throughout the course of this project. Her support and belief in me were instrumental in keeping me motivated and focused on my goals.

Additionally, I would like to thank my friends and colleagues for sharing memorable moments and our aspirations with me throughout my academic journey.

I would also like to acknowledge the contributions of the numerous researchers and professionals in the field, whose work provided valuable insights and resources.

Finally, I would like to extend my gratitude to my parents, Andreas and Fotini , Their selfless dedication to my growth and success has been a source of inspiration and I am forever grateful for the sacrifices they have made. I am deeply thankful for the family I have been blessed with.

Περιεχόμενα

ABSTRACT

Small open reading frames (sORFs) are proteins with a length of less than 100 amino acids, which were traditionally overlooked in genome annotation. However, recent studies have shown the importance of small proteins and their diverse functions in evolution. Current algorithms for the annotation of sORFs are not yet adequate, leading to a need for improved methods. This report presents a comprehensive literature review of the most recent and popular tools for the classification and annotation of sORFs. Additionally, combining the results of the tools, three individual classifiers were trained and evaluated for their performance. The results of the evaluation aim to provide an accurate and reliable solution for identifying sORFs.

ΠΕΡΙΛΗΨΗ

Τα μικρά ανοικτά πλαίσια ανάγνωσης (sORFs) είναι πρωτεΐνες με μήκος λιγότερο από 100 αμινοξέα, οι οποίες συνήθως δε λαμβάνονταν υπόψη στο σχολιασμό των γονιδιωμάτων. Ωστόσο, πρόσφατες έρευνες έχουν δείξει την σημασία των μικρών

πρωτεϊνών και της ποικιλίας των λειτουργών τους στην εξέλιξη. Οι τρέχοντες αλγόριθμοι για την αναπαράσταση των sORFs δεν είναι ακόμα επαρκείς, οδηγώντας σε μια ανάγκη για βελτιωμένες μεθόδους. Η εργασία αυτή παρουσιάζει μία εκτενή βιβλιογραφική έρευνα των πιο πρόσφατων και δημοφιλών εργαλείων για την ταξινόμηση και τον σχολιασμό των sORFs. Επιπλέον, συνδυάζοντας τα αποτελέσματα των εργαλείων, εκπαιδεύτηκαν και αξιολογήθηκαν τρεις ξεχωριστοί ταξινομητές για την απόδοση τους. Τα αποτελέσματα που προέκυψαν στοχεύουν σε μία περισσότερο ακριβή και αξιόπιστη λύση για το χαρακτηρισμό των sORFs.

Keywords

sORFs, coding , compared methods , noncoding , tools , evaluation, classifier

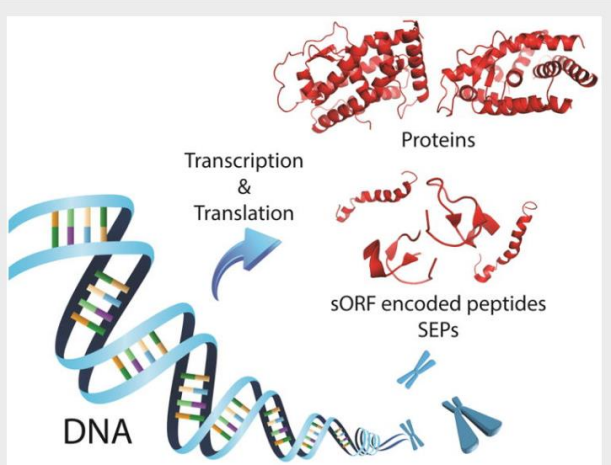

Figure 1: Protein and s(m)ORF-Encoded Polypeptides (SEPs) expression. (Chu et el., 2015)

INTRODUCTION

In general, the sequences of nucleotides between the start and stop codons are known as open reading frames (ORFs) (Sieber et al., 2018). Small open reading frames (sORFs) are a specific type of ORF that have fewer than 100 codons (Andrews et al., 2014). Research suggests that 1.5% of the genome may encode sORFs (Leong et al., 2022). These translatable sORFs have been found in many types of RNA, including previously identified ORF regions, 5' and 3' untranslated regions (UTRs) of messenger RNAs (mRNAs), and non-coding RNAs (ncRNAs), including long non-coding RNAs (lncRNAs). Thanks to advancements in sequencing and proteomics

technologies, combined with computational analysis, a significant number of translatable sORFs have been experimentally identified (Schlesinger et al., 2022).

The idea that sORFs could encode functional microproteins was first proposed several decades ago (Basrai et al., 1997). However, it is still challenging to determine which sORFs are coding and which are non-coding, so little is known about the functionality of most sORFs. Many protein identification techniques have a bias towards small proteins, and sORFs often have unusual start codons, making identification difficult (Chu et al., 2015). Despite these challenges, recent advancements in 'omics' methodologies have made significant progress in this field.

Although lncRNAs are not typically considered to encode proteins, some studies have shown that sORFs within some lncRNAs are translated into micropeptides with a median length of 23 amino acids (Hartford & Lal, 2020). These micropeptides can be classified into various categories, such as upstream ORFs (uORFs), intergenic ORFs, long non-coding ORFs (lncORFs), short coding sequences (CDSs), and short isoforms. Transcriptome analyses have shown that the majority of the human genome is transcribed, with only 2% annotated as a protein-coding region (Djebali, Davis, et al., 2012). Despite their small size, micropeptides are gaining attention for their critical roles in various biological activities.

High-throughput sequencing technology has allowed for the sequencing of genomes and transcriptomes from many species, yielding a vast amount of genetic information. Much effort is dedicated to understanding this information, primarily through the identification of functional genomic elements such as messenger RNAs (mRNAs) and long non-coding RNAs (lncRNAs). mRNA has been extensively studied for decades and is well represented in genetic databases. In contrast, lncRNAs, which are transcripts longer than 200 nucleotides that are not translated into proteins, have only recently been recognized for their role as regulators of gene expression and links to genetic diseases.

The main goal of annotating genomes and transcriptomes is to identify mRNAs and lncRNAs. Over the last two decades, the ENCODE and GENCODE projects have made significant progress in identifying and characterizing all functional elements of the human and mouse genomes, including mRNAs and lncRNAs, through

combination of sequencing data and manual curation. For non-model organisms, annotation of these elements typically relies solely on computational inference.

In the following sections, a theoretical background about microproteins and information about sORFs will be described. Bibliographic literature including information about the compared tools which predict the coding potential of sORFs, will also be presented, as well as information about the related databases for each tool. The installation and usage of the tools will be explained, and the results of their manual and artificial execution will be provided as well as comparisons between the tools. After combining the results of the tools, three individual classifiers were trained and evaluated for their performance and the results will be reviewed. Suitable diagrams and tables will be included, and finally, future work and conclusion will be discussed.

THEORITICAL BACKROUND

Microproteins

A microprotein or sORF-encoded protein (SEP) is made up of 100 amino acids and is encoded by a short open reading frame (sORF), which is made up of about 300 nucleotides. With ribosome profiling (RIBO-Seq), which revealed sORF-based transcripts at multiple genomic regions, sORFs were discovered to have coding potential while were previously ignored by genome annotation systems as meaningless noise.

Recognition of small proteins at the proteome level is difficult. Although mass spectrometry (MS) can offer direct proof of small proteins, it is highly dependent on existing libraries, which mostly focus on large proteins rather than small proteins. Because of their short length, small proteins lack protease cleavage sites. Small proteins, on the other hand, are normally in low amounts and are filtered through the enrichment process (Hsu & Benfey, 2018). Ribosome profiling which is also known with the terms Ribo- seq or ribosomal footprinting, is a more sensitive method for global identification of translation events utilizing deep sequencing of ribosome-protected mRNA fragments (RPFs) (Ingolia et a., 2009), allowing for the identification of translated ORFs and translation initiation sites (TISs), the spread of ribosomes on mRNA, and the pace of translating ribosomes (Weiss & Atkins, 2011).

Information Related to ORFs

There are many features which can affect the predictions of the coding potential of sORFs. Some of them are described in this section :

1. **ORF coverage** refers to the proportion of the genome that is covered by ORFs, which are contiguous sequences of nucleotides that have the potential to encode a protein. In other words, it measures how much of the genome is made up of regions that have the potential to encode functional proteins. The ORF coverage can be used to identify regions of the genome that are likely to encode functional proteins, and thus provide valuable insights into the biology and evolution of an organism.
2. **ORF length** refers to the length of an open reading frame. An open reading frame (ORF) is a continuous stretch of codons in a DNA or RNA sequence that has the potential to encode a functional protein. The length of an ORF is determined by the start and stop codons and is usually expressed in number of base pairs or amino acids. The length of an ORF can provide valuable information about the potential size and function of the encoded protein.
3. **Nucleotide bias** in ORF refers to the unequal representation of different nucleotides (A, T, C, and G) in a particular Open Reading Frame (ORF). This bias is due to the non-random distribution of codons, which can influence the expression and stability of the encoded protein. Different organisms can have different levels of nucleotide bias, which can affect the prediction of ORFs and their coding potential.
4. **Codon usage bias:** The preference of certain codons over others to encode the same amino acid.
5. **TIS** stands for "Translation Initiation Site". In molecular biology and genetics, a TIS refers to the specific location on a messenger RNA (mRNA) molecule where the ribosome starts to translate the mRNA into a protein. The TIS is determined by the presence of a start codon (AUG) that marks the beginning of the coding sequence, and it is the point at which the ribosome initiates translation of the mRNA into a protein.
6. **GC content,** the proportion of Guanine and Cytosine residues in the ORF sequence.

7. **The secondary structure** which is the three-dimensional arrangement of the ORF sequence, which can impact its stability and function,
8. **The conserved domains:** The presence of conserved domains in the ORF sequence, which can provide clues about its function.
9. **Orthology:** The presence of homologous ORFs in other species, which can indicate evolutionary relationships and functional conservation.
10. **Gene expression:** The level of gene expression can vary between different ORFs and can impact their function.
11. **Post-translational modifications:** The modification of the ORF sequence after translation, which can impact its stability and function.

LITERATURE RESEARCH

Comprehensive bibliographic research was conducted to identify the state-of-the-art tools and packages available for predicting and classifying sORFs between the years 2018-2022. The research resulted in the identification of 15 tools, including random-forest-model, csORF-finder, CircPrimer 2.0, OrfPP, smORFer, DeepCPP, AnABlast, RNAsamba, CPPred-sORF, ANNOgesic, CNIT, MiPepid, FSPP and CPAT.

To further narrow down the list of tools, certain criteria were applied, including the ability to take a transcript sequence as input, compatibility with the human genome, and software availability. Some of the tools were found to be only suitable for prokaryotic organisms, as smORFer (Refer to Table1) or lacked human genome evaluation, like ANNOgesic (Refer to Table1) and most of them were written in Python code, with a few available as online tools. After the filtering process, 7 tools were deemed suitable and these are the random-forest model, csORF_finder, DeepCPP, CPPred-sORF, RNAsamba, CNIT and MiPepid (Refer to Table1).

TOOL/ METHOD	Input	Year	Availability /Software	Species	Journal
1 RANDOM-FOREST TOOL	RNA/DNA sequence	2023	Online tool: https://www.sciencedirect.com/science/article/pii/S1046202322002481	Human	
2.csORF-finder	Query sequences to be predicted in FASTA format	2022	A web server can be accessed via https://github.com/mengzhanggggg/csORF-finder_webserver python 3.8	Homo sapiens	Briefings in Bioinformatics
3. Circprimer 2.0	For annotating circRNAs, circBase ID and genomic location are accepted data. For searching circRNA in circBase, circBase ID, genomic location and gene symbol (e.g. SDF4) are accepted data.	2022	easy-to-use software and freely available at www.bio-inf.cn. Java Runtime Environmen t(JRE	Suitable for human datasets	BMC Bioinformatics
4. OrfPP	the sequence of reference genome (the genome annotation file (in gtf format) and the nucleotide diversity calculated from genomic variants (in vcf format)	2022	Python -> pip install OrfPP==1.0 Available: https://pypi.org/project/OrfPP/1.0/	Wheat, cotton, suitable for Prediction of open reading frames (ORFs) including sORFs from populational genomic variants.	Briefings in Bioinformatics
5. DeepCPP	DNA/RNA transcript	2020	Python , Java	H. sapiens, M.musculus, D. rerio, D. melanogaster, Anopheles gambiae etc	Briefings in Bioinformatics
6.smORFer		2021	https://github.com/AlexanderBartholomaeus/Mi	Prokaryotic organisms	Nucleid Acid Research

			MB_ribosome_profiling https://github.com/AlexanderBartholomaeus/smORFer		
7.AnABlast	Nucleotide sequence	2019	http://www.bioinfocabd.upo.es/ab/ online tool	accurate prediction of intergenic sORFs in eukaryotic genomes.	Methods and Protocols
8.RNAsamba	a FASTA file containing transcript sequences	2019	https://rnasamba.lge.ibi.unicamp.br/ online tool and available for local installation written in Python 3 and Rust	Trained with human RNA sequences	Nucleic Acids Research (NAR) Genomics and Bioinformatics
9.CPPred, CPPred-sORF	RNA sequence	2019,2020	implemented in Python and freely available at http://www.rnabinding.com/CPPred-sORF/	H. sapiens, M.musculus, D. rerio, D. melanogaster, S. cerevisiae, etc	Nucleic Acids Research
10. ribotricer	Ribo-seq	2019	https://github.com/smithlabcode/ribotricer , python package.	Human datasets	Bioinformatics
11. ANNOgesic	ANNOgesic requires a specific set of input information, either as coverage information in wiggle format or alignments in binary alignment map (BAM) format.	2019	available under an open source license (ISCL) at : https://github.com/Sung-Huan/ANNOgesic , command-line tool	tested with several RNA-seq datasets from bacterial as well as from archaeal species.	Gigascience

12. CNIT	Accepts RNA transcripts in FASTA format	2019	http://cnit.noncode.org/CNIT/, as both a web server and a downloadable stand-alone	Human	Nucleic Acids Research
13. MiPepid	Fasta file with only DNA sequences (not RNA sequences or protein sequences)	2020	easy to use and available at https://github.com/MindAI/MiPepid, Python3	Designed for human genome	Bioinformatics
14. FSPP	Ribo-seq, RNA-seq and mass spectrometry data are input into FSPP, sequenced samples	2018	https://www.bioinfo.org/FSPP	Tested on human	Frontiers in genetics
15. CPAT	Accepts input sequences in either FASTA- or BED-formatted data files	2013	web interface, which allows users to submit sequences and receive the prediction results (http://lilab.research.bcm.edu/cpat/index.php) implemented in C and Python and is freely at : http://code.google.com/p/cpat/	Homo sapiens, M.Musculus,, etc	Nucleic acids research

Table1. This table presents a summary of crucial information about the tools including the type of input sequence they require, the year of publication, their accessibility, the software they utilize, their intended species and the journal in which they were published. (ref. [1] – [16]).

INFORMATION ABOUT THE TOOLS

RELATED DATABASES

The tools analyzed in this study gather data from several highly regarded databases. This section provides information on the specific databases utilized.

ENSEMBL

According to Birney et al., (2004) Ensembl is a bioinformatics project that organizes biological information based on huge genomic sequences. It is a comprehensive source of consistent automatic annotation of individual genomes as well as the synteny and orthology linkages between them. It is also a framework for integrating any biological data that can be mapped onto attributes derived from the genomic sequence. Ensembl is offered as an interactive Web site, a set of flat files, and a comprehensive, portable open source software system for processing genomes. All data is provided without restriction, and the code is publicly available.

NONCODE

NONCODE9 is a knowledge database that organizes and annotates noncoding RNAs (ncRNAs) (excluding rRNAs and tRNAs). The database had been upgraded to NONCODE v6.0 as of November 2020, and it contained lncRNAs from 39 different species: 16 animals (including mice) and 23 plants. NONCODE v6.0 contains 173,112 human lncRNAs and 131,974 mouse lncRNAs. Furthermore, the database provides annotations describing the relating expression profile, the exosome expression profile, functional information, information related to conservation and associations with diseases in addition to the basic descriptors for each lncRNA, such as were is located, strand, how many exons it has, length, and sequence (Zhao et al., 2016).

NCBI

The National Center for Biotechnology Information (NCBI) is a national repository of molecular biology data. The aim of the NCBI is to create new information

technologies that will aid in the understanding of basic molecular and genetic mechanisms that govern health and illness. More specifically, the NCBI has been tasked with developing automated systems for storage and analysis of knowledge about molecular biology, biochemistry, and genetics; supporting the use of such databases and software by the medical and scientific communities; trying to coordinate national and global efforts to gather biotechnology information; and conducting research into advanced techniques of computer-based processing information for analyzing the structure and function of biological systems (NCBI Resource Coordinators , 2016).

SmProt

SmProt (Structural Motifs in Proteins) is a database that contains information about structural motifs in proteins. Structural motifs are small, recurring units of protein structure that are found across different proteins. The database is used to identify and classify these motifs, and to provide information about the functions and relationships between different proteins. SmProt provides data on the sequence, structure, and function of structural motifs, as well as information on the interactions between structural motifs and other proteins. The database is updated regularly, and is a valuable resource for researchers in the fields of structural biology, protein science, and molecular biology. As a result the database also provides information on small proteins in different organisms as well as information on the sequence, structure, and functional annotations of small proteins. Additionally, it also provides information on the experimental evidence that support the presence and functional role of small proteins in different organisms. The database is a useful resource for researchers studying the biology and function of small proteins and for those who want to know more about their potential as drug targets (Hao et al., 2018)

sORFs.org

sORFs.org is a database that provides information and resources related to sORFs (small open reading frames), which are short coding sequences that have the potential to encode functional peptides or have non-coding functions. The database is designed to support sORF research by providing a comprehensive resource for sORF discovery, characterization, and functional analysis. Some of the features of the sORFs.org database include: annotation of sORFs from multiple species, including human,

mouse, and fruit fly, tools for sORF prediction, analysis, and functional annotation, a comprehensive collection of information on the discovery and characterization of sORFs, access to sORF-related publications, data, and tools. The information generated from ribosome profiling can be used to populate and validate sORF databases such as sORFs.orgs ORFs.org is a valuable resource for researchers and scientists working in the field of sORF research and peptidomics. By providing access to a wide range of information and resources, the database aims to facilitate the discovery and understanding of sORFs and their functions (Olexiouk et al., 2019).

TOOLS

csORF-finder

In 2022 Zhang et al., proposed the csORF-finder, a tool for the identification of confirmed small open reading frames (csORFs) in human, mouse, and fruit fly genomes The code is written in python 3.8 and the sORFs which where used for the training and testing datasets had length <= 300 nucleotides.

Datasets

The creators of csORF-finder collected experimentally validated csORFs from SmProt database as well as filtered non-csORFs from NONCODE, Ensembl and NCBI RefSeq databases.

Preprocessing, features and processing

The authors used CD-HIT (Fu et al., 2012), to remove redundant sequences, mapped the remaining sORFs onto transcripts, and split the sORFs into csORFs and non-csORFs datasets. They further filtered the non-csORFs using ribosome profiling evidence and split the csORFs and non-csORFs into three datasets based on their location.

To distinguish between csORFs and non-csORFs, the authors extracted both nucleotide and translated peptide sequence features of sORFs. Nucleotide-based

features included i-framed-3mer, i-framed-CKSNAP, i-framed-TDE, hexamer score, and ORF length. Peptide-based features were amino acid composition (AAC), amino acid pair composition (AAPC), and Composition, Transition, and Distribution (CTD). They also proposed a new nucleotide-based feature called i-framed-TDE.

The i-framed-3mer, i-framed-CKSNAP, and i-framed-TDE are nucleotide-based features that are used to distinguish between coding small open reading frames (csORFs) and non-coding small open reading frames (non-csORFs). The term "i-framed" refers to the reading frame used by the tool to analyze the sequence. In this case, the tool is reading the sequence in increments of three nucleotides (3-mer) to identify certain patterns that are characteristic of coding sequences.

The i-framed-CKSNAP and i-framed-TDE are specifically designed to capture codon usage patterns in the sequence, while the i-framed-3mer captures the frequency of specific 3-mer patterns. These features, along with the hexamer score, ORF length, and the peptide-based features (AAC, AAPC, and CTD), are used by the authors to build a prediction model for distinguishing between csORFs and non-csORFs.

The prediction model was constructed with the integration of LightGBM and the efficient CapsNet. The hyperparameters of LightGBM and CapsNet were tuned with Bayesian optimization. The authors performed a 10-fold cross-validation test to evaluate the prediction performance of the model and used independent tests. The features used for the model construction were i-framed-3mer, i-framed-CKSNAP, i-framed-TDE, hexamer score, ORF length, AAC, AAPC, and CTD.

In conclusion, the authors presented a comprehensive approach for the classification of sORFs into csORFs and non-csORFs. The integration of LightGBM and CapsNet, along with the use of both nucleotide and translated peptide sequence features, resulted in a robust prediction model with high accuracy.

MiPepid

In 2019, Zhu & Gribskov created a machine learning coding prediction tool which called MiPepid. The training data sources which were used was SmProt and traditional non coding RNAs. MiPepid based on python 3, supports human genome and takes as input a fasta file, strictly with DNA sequences. The results are placed in a csv file. The output file includes an ID which belongs to a sORF, the sORF's DNA sequence, the ID of the original DNA sequence in the input fasta file where this sORF is extracted, the 1-based starting position of this sORF on the original DNA sequence, the 1-based ending position of this sORF on the original DNA sequence , the predicted class of the sORF, either coding or noncoding and the probability that this sORF is in the assigned class.

MiPepid, developed for predicting micropeptides encoded by sORFs and utilized a logistic regression (LR) model with the 4-mer features extracted from sORFs sequences. The researchers suggest that the nucleotide patterns in the sequence hold the key to deciding whether a small ORF is translated.

The metrics used for the evaluation performance were accuracy and F1 score. For the parameter optimization the researchers chose 10 fold – cross validation to tune and chose the optimal L.R's hyperparameters , which are regularization strength and threshold.

CPPred and CPPred-sORF

In 2019 Tong & Liu proposed CPPred to predict the coding potential of RNA sequences. Later in 2020 Tony et al., 2020 suggested the improved tool CPPred-sORF which was based on CPPred, and its use was to predict the potential of protein-coding sORFs.

Datasets

CPPred uses two models: the first one (Human-Model) is trained on human data, and tested on various species including human, mouse, fruit fly, zebrafish, and S. cerevisiae. The second one (Integrated-Model) is built for integrated species including human, mouse, zebrafish, fruit fly, S. cerevisiae, nematode and thale cress. The

human coding RNAs and non-coding RNAs data were obtained from NCBI RefSeq and Ensembl database respectively. The data was randomly split into a training set (2/3 of data) and a testing set (1/3 of data). The testing set was made non-redundant with the training set using CD-hit with a threshold of 80% similarity to ensure independence between the test and training sets. The same conclusion was reached when using BLASTCLUST with various similarity thresholds. The performance of CPPred was evaluated on the testing set. To create an interesting and challenging testing set, sequences with ORF fragments less than 303 nucleotides in length were selected from the coding RNAs in Human-Testing and the same amount of comparable length non-coding RNAs from Human-Testing were selected.

Features and feature selection

The CPPred features are a set of features used to predict the coding potential of RNA sequences. The features are extracted from published scientific literature and are comprised of four features proposed by CPAT (ORF length, ORF coverage, Fickett score, and Hexamer score), two features derived from CPC2 (ORF integrity and isoelectric point), and Gravy and Instability. The Fickett score considers the frequency of the four nucleotides (A, T, G, C) in the RNA sequence and their position values, which reflect the degree of codon preference. The ORF length is the length of the maximum open reading frame, which starts with a start codon and ends with a stop codon (UGA, UAA, or UAG). The AUG is selected as the start codon in this study.

In addition to the above-mentioned features, 30 new features were added, which are CTD features. CTD stands for global transcript sequence descriptors and describes the transcript sequence, which contains four types of nucleotides. The nucleotide composition feature describes the percent composition of each nucleotide in the transcript sequence. The nucleotide transition feature describes the percent frequency of the conversion of four nucleotides between adjacent positions. The nucleotide distribution feature describes the relative position of each nucleotide along the transcript sequence at 0, 25%, 50%, 75%, and 100%. The CTD features are calculated using the sequence of the RNA transcript and are represented by a combination of letters (A, T, G, and C) and numbers (0, 1, 2, 3, and 4).

CPPred uses a feature selection method called mRMR-IFS to select the best subset of features from a set of 38 features that are used to predict coding and non-coding

RNAs. The exact list of these 38 features is not provided in the paper. The mRMR-IFS method selects features based on mutual information and minimal redundancy, and a 10-fold cross-validation is used to select the best subset of features from the training set. The final prediction model uses a Support Vector Machine (SVM) classifier with a radial basis function as the kernel and MCC as the optimization function to determine the best parameters (C and γ). The optimal values of C and γ are determined using grid search with different subsets of features.

RNAsamba

RNAsamba is a state-of-the-art tool developed in 2020, utilizing a unique neural network architecture. This tool accepts FASTA files containing transcript sequences as input, and is specifically designed for six organisms, including H. sapiens, M. musculus, D. rerio, D. melanogaster, C. elegans and A. thaliana. The tool is available both as an online web server and as a downloadable package for local installation. It is implemented in two languages, Python and Rust.

RNAsamba takes into account two sources of information about the RNA transcript: the whole sequence and the longest open reading frame (ORF) found within the transcript. The whole sequence information is processed through two independent stacked IGLOO units (N1 and N2) with different kernel sizes, and the ORF information is processed through four different layers that consider different properties of the ORF (protein sequence, nucleotide k-mer frequencies, amino acid frequency, and ORF length). The output of these two sources of information are then combined using an attention mechanism, which weights the contribution of each branch to the final output. During training, the algorithm optimizes the weights of the attention mechanism to maximize the accuracy of the classification. The final output is then fed through a dense layer with a softmax activation to compute the probabilities for each class (coding vs non-coding).

Datasets

The sequences which were used for the evaluation retrieved from RefSeq and Ensembl as well as mRNAs and lncRNAs that were filtered to retain only the ORF fragments that were <303 nucleotides. The training data of the specific tool came

from previous publications of some tools. The specific tool was trained with human RNA sequences and tested in multiple datasets (Mus musculus, Danio rerio, Drosophila melanogaster, Caenorhabditis elegans and Arabidopsis thaliana).

CNIT

According to Gua et al., (2019) CNIT constitutes a method for classifying small open reading frames (sORFs) in DNA sequences with length into either protein-coding or non-coding transcripts.

Datasets

It was built using a dataset of 19752 protein-coding and 19752 non-coding RNAs of human origin and 2588 coding and non-coding RNAs of Arabidopsis thaliana species. The dataset was obtained from RefSeq and Ensembl databases, and 70% of the total dataset was used for training and 30% for testing. The remaining 10 animal and 25 plant species were used for cross-species validation. The CNIT training and testing datasets can be accessed from the CNIT download page. The tool does not consider sORFs in the lncRNA data sets and compares its performance in identifying mRNAs with sORFs. To compare the performance of CNIT with other tools, it was evaluated using independent test datasets from the CPC2 datasets website. These datasets include human, mouse, zebrafish, fruitfly, worm, and Arabidopsis thaliana datasets and are considered high-quality.

Features and model construction

The construction of the CNIT model involved the creation of a comparison frequency matrix of adjoining nucleotide triplets (ANTs) based on the training dataset. Based on this matrix, a most-like coding domain sequence (MLCDS) was found in each reading frame, and the best MLCDS was used to construct the XGBoost models. A total of 67 features were used to construct the XGBoost models, including the MMLCDS score, the standard deviation of the MLCDS scores, the standard deviation of the MLCDS lengths, and the MMLCDS codon frequency.

DeepCPP

The DeepCPP was created in 2021 and constitutes a deep neural network tool. The specific tool uses minimum distribution similarity feature selection for RNA coding potential prediction and nucleotide bias information. It based on python 3 and takes as input a raw DNA sequence.

Features

DeepCPP employs a convolutional neural network (CNN) with two layers to classify the sequences into coding or non-coding categories. The model is trained on two different datasets, one for normal RNA sequences and the other for sORF type RNA sequences. The model uses two main features to make its predictions: nucleotide bias and discontinuous k-mer.

Nucleotide bias is a feature that represents the preference of nucleotides in codons before and after the start codon. The positions at −3, −2, −1, 4, 5, and 6 around the start codon are considered crucial for enhancing translation initiation and are considered in this feature. The results of this feature showed that it is an important feature for distinguishing sORF type RNA sequences, but its performance on normal RNA sequences is not significant.

Discontinuous k-mer is a feature that uses the k-mer approach to represent the sequences. A k-mer is a contiguous sequence of k nucleotides. In the discontinuous k-mer feature, k-mers are not required to be contiguous and can overlap. This feature is important for capturing the local patterns in the RNA sequences that are indicative of their coding potential.

In addition to these two main features, the model also employs other features such as maximum ORF length, ORF coverage, mean hexamer score, Fickett score, and 1-mer to make its predictions. These features are known to be useful in finding coding potential. The final predictions are made by taking the average scoring of three models and using a threshold of 0.5 to make the decision. The model's performance is evaluated using several criteria such as accuracy, sensitivity, specificity, F-score, and Matthew's correlation coefficient. Subsequently eight different types of features are initially extracted from the raw sequence, of which the 450 features referred to sORF data and they used as input feature subsets for the deep network. These feature subsets are sorted using their developed feature selection approach, MDS. The mDS method was applied on human training data and on a separate dataset, Dtest_H_sORF, which contains sORF type data. The reason for using Dtest_H_sORF was to identify features that were favored by sORF type data.

Datasets

The construction of models with coding potential prediction for many species, including humans, is the main purpose of the researchers. In addition to creating new human test datasets (normal and sORF type), training and test datasets (normal and sORF type) for vertebrates and insects from NCBI RefSeq and Ensembl databases, they adopted the same human training datasets and four test datasets (human testing, human-sORF-testing, mouse-testing, and mouse-sORF-testing) from CPPred.

Random forest-based model

This is the most recent method, proposed from Yu et al., in 2023.The researchers downloaded protein-coding sORFs data on human, and mouse and the data source was sORFs.org. They chose only the sORFs which encode protein sequences with length less than 100 amino acids. The sequences with no start codon or end codon as well as those with internal stop codons were excluded. To remove the redundancy they used the CD-HIT program at a cutoff of 80 %. and this dataset would be the positive one. The random forest model used in bioinformatics is a tool that predicts the coding potential of short open reading frames (sORFs) in DNA sequences. The model has three main types of features. The first one was z-curve theory.This theory looks at the frequencies of the four base pairs (A, T, G, and C) in different codon positions of the ORF. The model calculates the frequency of adjacent nucleotides in different codon positions and detects deviations in oligonucleotide composition to identify the coding potential of the ORF. The second one was K-mer and codon frequency.The frequency of K-mer (strings of length K) and codons is calculated by sliding a window along the DNA sequence. The window size and step are pre-defined, and the frequency of each sub-string is calculated based on the number of occurrences in the sequence. Finally the model uses features from a previous study (CPPred-sORF) that calculates 38-dimensional features from CPPred and 12 features from mRNN The combined features from these three sources are used to make predictions about the coding potential of sORFs in a given DNA sequence.

To validate its performance, the dataset was divided into N subsets using N-Fold Cross-Validation and evaluated using various performance indicators such as accuracy, precision, recall, F1-score, and Matthews correlation coefficient. The model

was also tested using the cross-species validation strategy where it was trained on three datasets and tested on a fourth, non-overlapping dataset. To optimize the model's parameters, a grid-search method was used to determine the best combination of three parameters: the number of trees, the maximum depth, and the minimum number for splitting a leaf node.

Tool / Method	Publication Year	Training data Source	Feature	Feature selection	Algorithm
MiPepid	2019	SmProt, NCBI RefSeq, Ensembl, NONCODE	4-mer	-	L.R
CPPred	2020	H. sapiens, M. musculus, D. rerio, D. melanogaster, S. cerevisiae, C. elegans and A. thaliana	ORF length and coverage, Fickeet score, Hexamer score, ORF integrity, isoelectric point, gravy, instability, global transcript sequence descriptors	mRMR-IFS	SVM
RNAsamba	2020	Previous publications: CPC2, FEELnc, and mRNN	Nucleotide IGLOO, protein IGLOO, nucleotide {2,3,4}-mers, AAC, ORF length	-	IGLOO
CNIT	2021		MLCDs , standard deviation 6 MLCDS score values, standard deviation τωv 6 MLCDS lengths, 64 nucleotide triplets (NTs)	-	XGBoost
DeepCPP	2021	H. sapiens, M. musculus, D. rerio, D. melanogaster, Anopheles gambiae and Aedes aegypti	Maximum ORF length and ORF coverage, Fickeet score, Hexamer score, Nucleotide bias,1-mer, 3-mer for maximum ORF, 2-gap and 3-gap	mDS	CNN
csORF-finder	2022	SmProt, NCBI RefSeq, Ensembl, NONCODE	i-framed-3mer, i-framed- CKSNAP, i-framed-TDE, hexamer score, ORF length, AAC, AAPC, and CTD	LightGBM-IFS	Efficient-CapsNet and LightGBM
Random-forest model	2023	sORFs.org	z-curve, codon frequency, k-mer and reference from CPPred-sORF	-	RF

Table 2. This table collects all the information about the publication year, training data sources, features, feature selection and the algorithms of the compared tools (Zhang et al., 2022)

INSTALLATION OF THE TOOLS

Initially, the process of installing the various tools was carried out, followed by an attempt to run them using the information provided in their respective folders. Subsequently, each tool was executed with the relevant datasets.

csORF-finder

the csORF-finder-main package was first downloaded from GitHub and an environment was created in Anaconda named csorfinder. The necessary python modules, as specified in the GitHub repository, were then installed, including cudatoolkit, cudnn, tensorflow-gpu, protobuf, keras, lightgbm, pandas, and openpyxl. However, during the installation process, a conflict arose between packages and the installation of pandas version 1.4 was required for successful installation. Finally, the csORF_finder_predict_sORFs.py script was run, utilizing the input file provided in the GitHub folder and an empty csv file created for storing the results. The tool ran without any issues and provided the expected output (Zhang et al., 2022).

MiPepid

In this work, I also encountered challenges while executing the code for the sORF classification tool. The library dependencies were not specified clearly in the documentation, leading to difficulties in downloading the correct versions of the required packages. After thorough research, |I found the best combination of packages, including scikit-learn 0.22.2.post1, pandas 1.5.3, biopython 1.80, numpy 1.23 and python 3. The code was executed successfully with the specified command line argument and generated the expected output file, which contained information such as the sORF ID, DNA sequence, original DNA sequence ID, start and end positions, predicted class, and class probability.

CNIT

Despite being accessible as a mobile-friendly web server for investigation, the tool is also available as a downloadable stand-alone package. However, I encountered a technical issue with the specific online tool which is available at http://cnit.noncode.org/CNIT/ , as during my attempt to use it, the browser did not open and as a result, I was unable to utilize the tool for my analysis.

RNAsamba

The installation of RNAsamba was straightforward and no compatibility issues were encountered. This tool is equipped with a deep learning architecture, utilizing TensorFlow and Keras libraries. RNAsamba offers both training and classification commands, with the latter being used in this study to evaluate the coding potential of transcripts contained in an input FASTA file and classify them as coding or non-coding.

CPPred-sORF

The installation of the specific tool posed a significant challenge due to the limited information provided in the documentation. The only information provided was to install the biopython package using the pip install biopython==1.75 –user command. However, the code is written in Python 2.7, which made it difficult to find a compatible version of biopython. After extensive research, it was found a compatible version of biopython, but the process was time-consuming. Furthermore, I had to find compatible versions of numpy, scikit, and cython packages to avoid conflicts. Despite the efforts, the code still produced errors, as it was written in an outdated programming language. The installation process consumed a significant amount of time and resources, highlighting the importance of clear and detailed documentation for software tools.

DeepCPP

The documentation lacked information about the specific versions of the required libraries, including numpy, pandas, csv, keras, Bio, and math. Despite this, the code

was written in Jupyter and an error message was encountered regarding library imports. To resolve this issue, a new environment was created using the deepCPP_env.yaml file that contained all the necessary dependencies. The new files, utils_ IncRNA.py and DeepCPP_n.py, were copied to the code directory, and the command "python deepCPP_n.py" was executed, resulting in a successful run of the code.

The random forest-basted model in an online tool available http://guolab.whu.edu.cn/codingCapacity/index.html, so it didn't occur any issue.

Taking all the above mentioned, the conflicts of packages in each tool can cause significant delays in the work process. It is imperative that the creators of each tool provide strict specifications of the required package versions to prevent such conflicts from occurring. This will not only save time but also ensure that the tool runs smoothly and produces the desired results. It is crucial for the creators of such tools to specify the correct library dependencies and their versions to avoid such conflicts and save time in the execution process.

EVALUATION OF THE TOOLS

Evaluation Datasets

Subsequently, the tools were executed using data from two datasets to be evaluated. The first dataset consisted of validated, labeled protein coding sORFs obtained from the Ensembl database and referred to as the positive dataset. The second dataset comprised of validated noncoding IDs, which were obtained from intronics and referred to as the negative dataset.

Evaluation Criteria

In this assignment, were used some significant evaluation criteria mentioned which are crucial to be analyzed.

Confusion matrix

The confusion matrix is a performance metric commonly used in machine learning to evaluate the quality of a classification model. It is a table that summarizes the number of correct and incorrect predictions made by the model on a set of test data.

The confusion matrix calculates the following values:

True positives (TP): the number of positive instances that were correctly classified as positive by the model.

False positives (FP): the number of negative instances that were incorrectly classified as positive by the model.

True negatives (TN): the number of negative instances that were correctly classified as negative by the model.

False negatives (FN): the number of positive instances that were incorrectly classified as negative by the model.

The values in the confusion matrix are used to calculate various performance metrics, such as accuracy, precision, recall, and F1 score, which can help assess the quality of the classification model. The confusion matrix provides a more detailed and informative analysis of the model's performance compared to simple metrics like accuracy, especially when the classes are imbalanced or the cost of false positives and false negatives are different.

Precision

An evaluation of a model's precision in predicting true labels is known as precision. The answer to the inquiry, "How often was a model's positive prediction accurate?" is precision. The proportion of the results that are significant is known as precision A high precision indicates that a test or a model produces very few false positive results among all positive results. Precision constitutes an important metrics to consider when evaluating the performance of a binary classification system, as they provide

complementary information about the ability of the system to accurately identify positive and negative samples. It is calculated by the formula:

$$\frac{TP}{TP+FP}$$

where TP are the true positive and FN the false positive results.

Sensitivity

Sensitivity of Recall is a metric used to evaluate the performance of a classification model, particularly in binary classification problems. It measures the proportion of actual positive cases that were correctly identified as such by the model. Recall is also known as the sensitivity or the true positive rate (TPR).

In a binary classification problem, there are four possible outcomes for each prediction: true positive (TP), false positive (FP), true negative (TN), and false negative (FN). Recall is defined as the ratio of true positive predictions to the total number of actual positive cases.

In other words, recall gives the information on what fraction of the positive cases the model was able to identify correctly. A high recall value indicates that the model has a low false negative rate, which is important in applications where missing a positive case is particularly costly (e.g. in medical diagnosis). It is calculated by the formula :

$$\frac{TP}{TP+FN},$$

where TP are the true positive and FN the false negative results.

Specificity

Specificity (true negative rate) refers to the probability of a negative test, conditioned on truly being negative. True negative rate (TNR), also known as the specificity of a test, is the percentage of samples that are negative and provide a negative result when the test in question is used (Yao et al., 2019). Like precision, specificity is an

important metrics to consider when evaluating the performance of a binary classification system. It is calculated by the formula :

$$\frac{TN}{TN+FP},$$

where TN are the true negative and FP the false positive results.

Accuracy

Accuracy is a measure of how well a model or system performs in terms of correctly predicting the output. It is the fraction of correct predictions made by the model over the total number of predictions. In other words, accuracy is a measure of how close the model's predicted results are to the actual results.

For example, in a binary classification problem, accuracy is calculated as the ratio of correctly classified positive examples and negative examples to the total number of examples. The accuracy metric is widely used to evaluate the performance of machine learning models, and it is particularly useful for tasks where the target variable is binary (e.g. yes/no, true/false). Accuracy is calculated by the formula :

$$\frac{TP+TN}{total\ number\ of\ samples},$$

It is important to note that accuracy is not always the best measure of model performance, as it does not always reflect the trade-off between false positives and false negatives. In some cases, a more appropriate evaluation metric might be precision, recall, F1 score, or a confusion matrix.

F1 score

The F1 Score is a measure of a model's accuracy that combines precision and recall. It is calculated as the harmonic mean of precision and recall and provides a single number that summarizes the balance between precision and recall for a classifier.

The formula for F1 Score is given as:

$$2x\ \frac{precision\ x\ recall}{precision+recall},$$

A high F1 score indicates that the model has a good balance of precision and recall, while a low F1 score indicates that the model has a poor balance of precision and recall. The F1 score is particularly useful when there is an uneven distribution of class in the dataset, as it provides a more balanced measure of accuracy than precision or recall alone.

RESULTS

MANUALLY

The results which came after running the tools with the above mentioned datasets showed that RNAsamba found 33 TP (true positive/coding), 88 FP (false positive/noncoding), 847 TN (true negative/true noncoding), and 47 FN (false negative/false noncoding) and as a result noticed 0,4125 sensitivity, 0,9 specificity and 0,272 precision (refer to Table3). DeepCPP found 30 TP , 32 FP , 903 TN and 50 FN and as a result its sensitivity, specificity and precision were 0,375, 0,965 and 0,483 respectively. Finally csORF-finder obtained 69TP , 283 FP, 652 TN and 11 FN and as a result 0,8625 sensitivity , 0,697 specificity and 0,196 precision. Finally,

Random-Forest obtained 75 TP, 816 FP, 119 TN and 5FN and as a result 0,937 , 0,127 Specificity and 0,084 Precision (refer to Table3).

MiPepid produced more results than the inputs provided. Specifically, it generated 208 results (80 initial) for the Ensembl labeled dataset, of which 137 were duplicates, and 966 (instead of the initial 935) for the Intronics dataset. To address this, I removed the duplicates by retaining only one coding result from the positive dataset and one non-coding result from the negative dataset for each duplicate. However, it appeared that the tool had lost some inputs in the process. As a result, further research is necessary to understand why such results were generated and the results of this tool were not considered.

TOOL / METHOD	TP (TRUE POSITIVE/TRUE CODING)	FN (FALSE NEGATIVE/FALSE NONCODING	TN (TRUE NEGATIVE/TRUE NONCODING)	FP (FALSE POSITIVE/FALSE CODING)	SENSITIVITY	SPECIFICITY	PRECISION
RNAsamba	33	47	847	88	0,4125	0,9	0,272
DeepCPP	30	50	903	32	0,375	0,965	0,483
csORF-finder	69	11	652	283	0,8625	0,697	0,196
RANDOM-FOREST MODEL	75	5	119	816	0,937	0,127	0,084

Table 3. This table displays the TP (true positive/true coding), FN (false negative/false noncoding), TN(true negative/true noncoding), FP(false positive/false coding), sensitivity, specificity, and precision values obtained from the various methods when they were run using validated protein coding and non-coding sORFs as inputs.

ARTIFICIALLY

Even if the input is the same, the artificial and manual results of a tool could vary. There could be a variety of reasons for this, including differences in the configuration of the tool, version, or setup, the method or parameters utilized, or even human mistake. As a result, the next step was to run the tools artificially to compare them with the results that came up from the manual execution. All the artificial results were the same with those which came up after running the tool manually.

COMPARISONS

Based on the current assignment:

Based on the results obtained from the evaluation of the tools using the mentioned datasets, some conclusions were reached.

When comparing the results of running the tools on the datasets, it can be concluded that the performance of the tools varied in terms of sensitivity, specificity and precision. RNAsamba had a low sensitivity (0.4125) (refer to Table3), meaning that it was not able to detect all the true coding sequences. Its specificity was high (0.9), which indicates that it was able to accurately identify noncoding sequences. However, its precision was low (0.272), meaning that a lot of false positive coding sequences were detected (see Fig. 2).

DeepCPP had the lowest sensitivity (0.375) compared to the other tools, meaning that it couldn't detect the true coding sequences. Its specificity was very high (0.965) and its precision was moderate (0.483) (see Fig. 2).

Moreover, csORF-finder had the highest sensitivity from the above-mentioned tools and the second highest of all the tools (0.8625), indicating that it was able to detect the majority of true coding sequences. Its specificity was moderate (0.697), and its precision was low (0.196), meaning that a large number of false positive coding sequences were detected (see Fig. 2).

Finally, Random-Forest had the highest sensitivity (0.937) and the lowest specificity (0.127) and precision (0.084). This means that it was able to detect a large number of true coding sequences, but also a lot of false positive coding sequences were detected, making the results less reliable (see Fig. 2).

In conclusion, the results suggest that RF is the best at identifying coding ORFs (high sensitivity), while DeepCPP is the best at correctly identifying non-coding ORFs (high specificity). On the other hand, precision indicates that DeepCPP had the best overall performance, with the highest precision among the four tools.

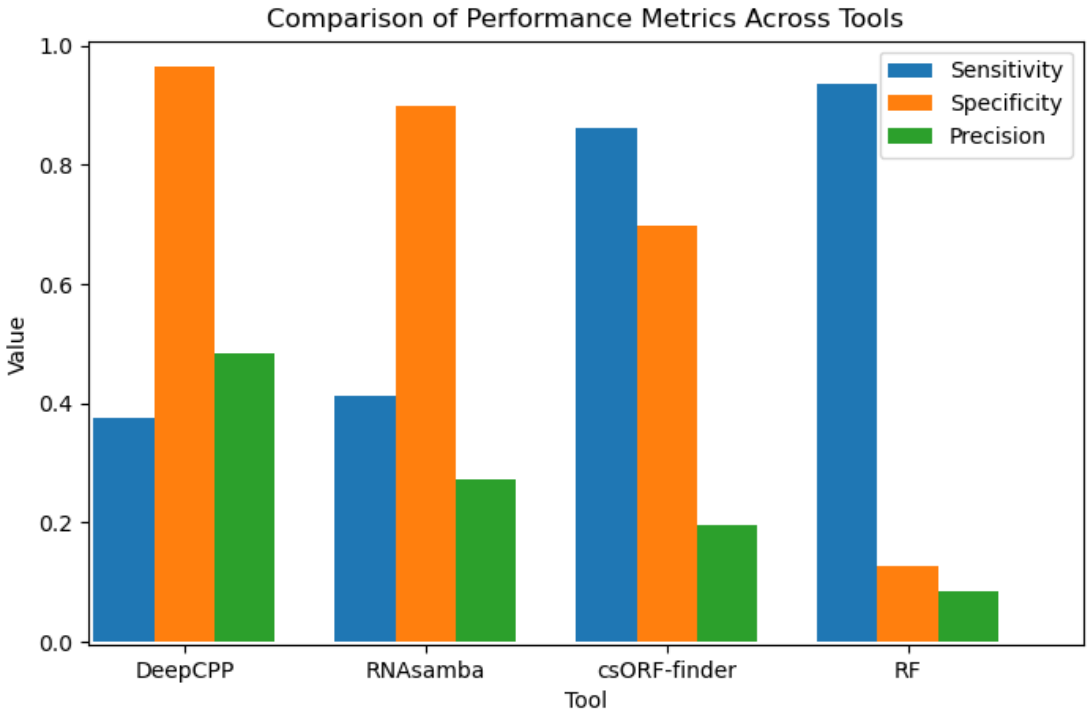

Figure 2. This diagram is a bar chart that compares the performance metrics of the four different tools: DeepCPP, RNAsamba, csORF-finder, and RF. The three-performance metrics that are being compared are sensitivity, specificity, and precision, with each metric represented by a different color bar. The bars for each metric are grouped together for each tool. The x-axis shows the name of each tool, while the y-axis shows the value of the performance metric. The height of each bar represents the value of the corresponding performance metric for a given tool. The yellow bar in DeepCPP shows high specificity, indicating that this tool is able to accurately identify true negatives (i.e. non-coding sequences) with a low rate of false positives (i.e. misclassifying coding sequences as non-coding). However it seems to have the highest precision according to its green bar. The blue bar in RNAsamba shows moderate sensitivity, meaning that this tool has a reasonable ability to detect true positives (i.e. coding sequences) but may miss some coding regions and produce false negatives. The green bar in csORF-finder shows relatively low specificity, indicating that this tool may mistakenly identify non-coding sequences as coding, resulting in a higher rate of false positives. The green bar in RF shows low precision, meaning that this tool is likely to produce many false positives when identifying coding sequences. The blue bar in RF shows that it has the highest sensitivity , however the yellow and green bar show that it has the lowest precision and specificity.

Based on the compared tools' papers

In their study Zhang et al. (2022) which are the creators of csORF-finder compared the performance of csORF-finder with four other tools, RNAsamba, CPPred, MiPepid, and DeepCPP, using three species as test subjects: H. sapiens, Mus musculus, and D. melanogaster.

The results of the study showed that DeepCPP and CPPred produced lower sensitivity values compared to csORF-finder when applied to the nonCDS-sORFs datasets of H. sapiens and D. melanogaster. This means that both DeepCPP and CPPred were less

effective in detecting coding signals in the UTR sequences of the two species compared to csORF-finder. However, DeepCPP and CPPred had higher specificity values compared to csORF-finder, meaning they were better at avoiding false positives.

The study also found that RNAsamba outperformed MiPepid in predicting sORFs, indicating its superiority over the latter tool. These findings demonstrate the need for further improvements in the prediction tools used for sORF detection, especially in detecting coding signals in UTR sequences.

Zhang et al. (2021) which conducted the DeepCPP's study found that DeepCPP performed better than CPPred. The results from the study showed that DeepCPP gave higher performances than CPPred on four different test datasets, and its overall performance from all types of measurements was enhanced, particularly for sORF RNA datasets.

The researchers also noted that there is room for improvement in the species-specific model of RNAsamba, and suggested that its performance could be further enhanced. The study found that DeepCPP demonstrated high predictive power, with 98.20% for human data, 95.78% for vertebrate data, and 96.16% for insect data. Additionally, DeepCPP showed at least 1.08%, 17.09%, and 3.14% improvement in accuracy compared to previous approaches for human, vertebrate, and insect data, respectively.

In accordance with Zhang et al. (2022), the study found that the sensitivity values of DeepCPP and CPPred were significantly lower compared to csORF-finder when applied to the H. sapiens and D. melanogaster non-CDS-sORFs datasets. These findings suggest that there is still room for improvement in the tools used for sORF detection.

CODE IMPLEMENTATION

Based on the performance metrics, it seems like none of the bioinformatics tools are performing very well on their own. The average sensitivity (true positive rate) ranges from 0.375 to 0.8625, which means that the tools are detecting coding sORFs with an accuracy ranging from 37.5% to 86.25%. The average specificity (true negative rate) ranges from 0.127 to 0.965, meaning that the tools are correctly classifying non-

coding sORFs with an accuracy ranging from 12.7% to 96.5%. The average precision (positive predictive value) ranges from 0.084 to 0.483, meaning that the tools are making correct predictions about coding sORFs with an accuracy ranging from 8.4% to 48.3%.

In this assignment is represented a program which includes three classifiers (Decision Tree, K-Nearest Neighbors, and Random Forest) which were used to make independent predictions on the merged data came from the outputs of the CSVs files of the above-mentioned tools. Each classifier produced its own set of predictions for the given dataset. The performance of each individual classifier was evaluated on both the training and testing sets using various metrics and the best prediction was selected.

Pre-processing

The goal of preprocessing is to clean and prepare the data for analysis, and converting the classifications into binary values is a step towards transforming the data into a usable format for the classifier. The preprocessing steps involved merging multiple outputs from different tools into a single dataframe and cleaning it up for further analysis. This involved some modifications to the input data to make it compatible with the classifier. Firstly I removed the "[“" from the ‘SorfID’ column in the csORF’s finder results. I also selected the basic columns that were common among all the tools and dropped the ‘coding score’ column, which was present only in the RNAsamba files, since it was showed with NaN values in the merged dataframe and was deemed not essential. Additionally, I replaced variations of "coding" and "noncoding" in the "classification" column with a standard representation of "coding" and "noncoding". Subsequently it was performed a binary classification which involved the mapping of the text-based predictions ('coding' or 'noncoding') to binary values (0 or 1) and removing the duplicate rows based on the combination of “SorfID” and “classification” columns. Keeping multiple non-identical predictions for the same input can help to improve the training of the classifiers because it provides more diverse information for the training data. This can potentially increase the accuracy and robustness of the classifiers. On the other hand, keeping only identical predictions may not provide as much useful information for the training data. Finally I filled the NaN values in the “Probability” column with the mean of the noNaN values and replace the NaN in “SorfAnnotation” with the most common value , as thy were

only 5 and after comparing with the initial file I realized that these values were common with common values.

The true labeled data was processed by putting the true labels (TP and TN) into one file, which is then merged with the final dataframe of the tools. The merging was done based on the SorfID column. The true labels were also processed into a binary format, with TP and TN being mapped to the binary values 0 and 1, respectively. The final merged data contains both the predictions of the individual tools and the true labeled binary values for each SorfID. The final preprocessed data was then saved as a CSV file.

Method

The method included the training and evaluation of three individual classifiers (DecisionTreeClassifier, KNeighborsClassifier, and RandomForestClassifier) on a binary classification problem. The data used for the analysis was preprocessed and merged from multiple sources and was stored in the 'merged_data.csv' file. The input features were one-hot encoded to convert categorical variables into numerical form. The dataset was split into 50-50 training and testing datasets. The three individual classifiers were trained on the training dataset, and predictions were made on the test dataset. The training set included 891 negative samples and 68 positive samples while the testing set 892 negatives and 67 positives.

CLASSIFIERS' RESULTS

The results from the training set show that the DecisionTree and RandomForest classifiers performed perfectly with 100% sensitivity, F1 score and precision, correctly identifying all 68 positive cases in the dataset (see Table 4). The KNeighbors classifier also achieved high accuracy with 91.2% sensitivity, 94.0% F1 score, and 96.9% precision. However, it had a higher false negative rate compared to the other classifiers, misclassifying 6 positive cases as negative (see Table 4.).

Based on the testing set's results, it seems that all the classifiers are performing very well. The DecisionTreeClassifier achieved perfect sensitivity (1.000), meaning that it correctly identified all the true positives in the dataset. The RandomForestClassifier and the had a sensitivity of 0.955, indicating that they correctly identified 95.5% of the true positives. The KNeighborsClassifier had a lower sensitivity of 0.880, meaning that it correctly identified only 88% of the true positives (see Table 5).

In terms of precision, the DecisionTreeClassifier and RandomForestClassifier achieved perfect precision of 1.000, meaning that all their predicted positive cases were true positives. The KNeighborsClassifier had a slightly lower precision of 0.970, indicating that it had a higher rate of false positives in its predictions (see Table 5).

Overall, while all the classifiers performed well, the DecisionTreeClassifier achieved the highest sensitivity and precision values, making it a strong choice for this dataset.

However, if the class distribution is highly imbalanced, as it happens in this case (68 positives and 891 negatives) the model is likely to predict the majority class most of the time, leading to poor performance for the minority class. This is because the model is optimized for the majority class, but it may not generalize well to the minority class.

It is worth mentioning that there are metrics that are more sensitive to class imbalance, such as F1 score. Unlike accuracy, the F1 score considers both precision and recall, making it a better metric for imbalanced datasets where it is possible for a classifier to achieve high accuracy by simply predicting the majority class for all samples. The F1 score is the harmonic mean of precision and recall, giving equal weight to both precision and recall, making it a good metric for datasets with imbalanced class distributions. After calculating the F1 score the metrics had also high values. Such good performance is indicating that the model has a good balance between precision and recall.

Decision Tree classifier outperforms the other classifiers.

The results of the classifiers show that the DecisionTree outperformed the other classifiers in terms of sensitivity, F1 score and precision on the binary classification problem (see Table 4). The DecisionTree achieved a sensitivity score of 1.000,

meaning it correctly identified all of the true positives in the dataset. In addition, the DecisionTree achieved perfect precision of 1.000, indicating that all its predicted positive cases were true positives (see Table 5). Moreover, it noticed F1 score of 1.000 which indicates that it has an excellent balance between precision and recall (see Table 5).

In contrast, the KNeighborsClassifier had the lowest sensitivity of 0.880, indicating that it correctly identified only 88% of the true positives. Moreover, the KNeighborsClassifier had a slightly lower precision of 0.970, meaning it had a higher rate of false positives in its predictions.

Overall, although each classifier noticed very good scores, the high scores of the Decision Tree make it a strong choice for this dataset. However, as it mentioned before it is important to note that the class distribution was highly imbalanced, and therefore, the model may not generalize well to the minority class.

Thresholds

Subsequently, I used different thresholds to evaluate the performance of the DecisionTree classifier. By varying the threshold values, I aimed to investigate the effects of different decision thresholds on the sensitivity, precision, and F1 score of the classifier.

The classifier achieved an impressive 100% accuracy in all cases, which suggests that it is highly effective in correctly classifying the target variable (see Table 6).

When a threshold of 0.8 was used, the classifier correctly identified 67 true positive cases and 0 false negative cases. This resulted in a sensitivity score of 1, indicating that the classifier was able to correctly identify all positive cases. The F1 score and precision were also perfect at 1, indicating that the classifier was highly precise in its predictions (see Table 6). Similarly, when a threshold of 0.5 was used, the classifier again achieved a perfect sensitivity, F1 score, and precision score, while correctly identifying the same 67 true positive cases and 0 false negatives. However, the number of false positives was also zero, indicating that the classifier was highly selective in its predictions (see Table 6).

Overall, these results suggest that the DecisionTree classifier is highly effective in classifying the target variable with high precision and sensitivity . The use of different threshold values did not seem to have a significant impact on the performance of the classifier, which remained consistently accurate across all cases. These results demonstrate the potential of DecisionTree classifiers for accurately classifying complex datasets and highlight the importance of carefully selecting evaluation metrics to ensure that the model's performance is accurately assessed.

Table 4. This table shows the performance metrics of three different classifiers: DecisionTree, RandomForest and KNeighbors, It shows the true positive (TP), false negative (FN), true negative (TN), false positive (FP) values, as well as the sensitivity, F1 score, and precision of each classifier according to the training set's results.

Classifier	**TP (True positive)**	**FN (False negative)**	**TN (True negative)**	**FP (False positive)**	**Sensitivity**	**F1 SCORE**	**precision**
DecisionTree	68	0	891	0	1	1	1
RandomForest	68	0	891	0	1	1	1
KNeighbors	62	6	889	2	0.912	0.940	0.969

Table 5. This table shows the performance metrics of three different classifiers: DecisionTree, RandomForest and KNeighbors, It shows the true positive (TP), false negative (FN), true negative (TN), false positive (FP) values, as well as the sensitivity, F1 score, and precision of each classifier according to the testing set's results.

Classifier	**TP (True positive)**	**FN (False negative)**	**TN (True negative)**	**FP (False positive)**	**Sensitivity**	**F1 SCORE**	**precision**
DecisionTree	67	0	892	0	1	1	1
RandomForest	64	3	892	0	0.955	0.977	1
KNeighbors	59	8	890	2	0.88	0.924	0.97

THRESHOLD	TP (TRUE POSITIVE)	FN (FALSE NEGATIVE]	TN (TRUE NEGATIVE)	FP (FALSE POSITIVE)	SENSITIVITY	F1 SCORE	PRECISION
0.8	67	0	892	0	1	1	1
0.5	67	0	892	0	1	1	1
-	67	0	892	0	1	1	1

Table 6. This table shows the results of the metrics of sensitivity, f1-score and precision after the evaluation of the DecisionTree classifier in 3 cases, two with threshold (0.8, 0.5) and one without threshold.

FUTURE WORK

Despite the impressive results obtained from the current classifier, it is important to note that the dataset used in this study is imbalanced, which may have influenced the performance of the classifier. Therefore, it is crucial to conduct further examination to ensure that the results are robust and generalizable to new data. To achieve this, several future work options could be considered. Firstly, experimenting with different techniques such as weighting or stacking could help to improve the performance of the classifier. Secondly, evaluating the performance of the classifier on a larger, more diverse dataset would provide a better understanding of the generalizability of the results. Thirdly, incorporating more features from the tools into the classifier could further reinforce its performance. These future work options would provide valuable insights into the robustness and generalizability of the classifier and would help to ensure that the results are reliable and accurate.

CONCLUSION

In conclusion, the classification and annotation of small open reading frames (sORFs) is a crucial task in understanding the diverse functions of small proteins in evolution. This report provides a comprehensive literature review of the most recent and popular

tools for sORF classification and annotation. The evaluation of these tools aim to provide a more accurate and reliable solution for the identification of sORFs. The results show that the DecisionTreeClassifier provides impressive results, however, it is important to note that the dataset used in the evaluation is imbalanced and further examination is needed. The future work includes experimenting with different techniques, evaluating the performance on a larger and more diverse dataset and incorporating more features of the tools which where used. Overall, this report provides valuable insights and information for researchers and practitioners in the field of sORF classification and annotation.

REFERENCES

[1] Rubio, A., Casimiro-Soriguer, C. S., Mier, P., Andrade-Navarro, M. A., Garzón, A., Jimenez, J., & Pérez-Pulido, A. J. (2019). AnABlast: re-searching for protein-coding sequences in genomic regions. Gene Prediction: Methods and Protocols, 207-214.

[2]Wang, L., Park, H. J., Dasari, S., Wang, S., Kocher, J. P., & Li, W. (2013). CPAT: Coding-Potential Assessment Tool using an alignment-free logistic regression model. Nucleic acids research, 41(6), e74-e74.

[3] Li, H., Xiao, L., Zhang, L., Wu, J., Wei, B., Sun, N., & Zhao, Y. (2018). FSPP: a tool for genome-wide prediction of smORF-encoded peptides and their functions. Frontiers in genetics, 9, 96.

[4] Choudhary, S., Li, W., & D. Smith, A. (2020). Accurate detection of short and long active ORFs using Ribo-seq data. Bioinformatics, 36(7), 2053-2059.

[5] Zhang, M., Zhao, J., Li, C., Ge, F., Wu, J., Jiang, B., ... & Song, X. (2022). csORF-finder: an effective ensemble learning framework for accurate identification of multi-species coding short open reading frames. Briefings in Bioinformatics, 23(6), bbac392.

[6] Zhang, Y., Jia, C., Fullwood, M. J., & Kwoh, C. K. (2021). DeepCPP: a deep neural network based on nucleotide bias information and minimum distribution

similarity feature selection for RNA coding potential prediction. Briefings in bioinformatics, 22(2), 2073-2084.

[7] Camargo, A. P., Sourkov, V., Pereira, G. A. G., & Carazzolle, M. F. (2020). RNAsamba: neural network-based assessment of the protein-coding potential of RNA sequences. NAR genomics and bioinformatics, 2(1), lqz024.

[8] Tong, X., Hong, X., Xie, J., & Liu, S. (2020). Cppred-sorf: Coding potential prediction of sorf based on non-aug. BioRxiv.

[9]Tong, X., & Liu, S. (2019). CPPred: coding potential prediction based on the global description of RNA sequence. Nucleic acids research, 47(8), e43-e43.

[10] Zhu, M., & Gribskov, M. (2019). MiPepid: MicroPeptide identification tool using machine learning. BMC bioinformatics, 20(1), 1-11.

[11] Guo, J. C., Fang, S. S., Wu, Y., Zhang, J. H., Chen, Y., Liu, J., ... & Zhao, Y. (2019). CNIT: a fast and accurate web tool for identifying protein-coding and long non-coding transcripts based on intrinsic sequence composition. Nucleic acids research, 47(W1), W516-W522.

[12] Yu, S. H., Vogel, J., & Förstner, K. U. (2018). ANNOgesic: a Swiss army knife for the RNA-seq based annotation of bacterial/archaeal genomes. Gigascience, 7(9), giy096.

[13] Bartholomäus, A., Kolte, B., Mustafayeva, A., Goebel, I., Fuchs, S., Benndorf, D., ... & Ignatova, Z. (2021). smORFer: a modular algorithm to detect small ORFs in prokaryotes. Nucleic Acids Research, 49(15), e89-e89.

[14] Yu, J., Jiang, W., Zhu, S. B., Liao, Z., Dou, X., Liu, J., ... & Dong, C. (2023). Prediction of protein-coding small ORFs in multi-species using integrated sequence-derived features and the random forest model. Methods.

[15] Jiang, M., Ning, W., Wu, S., Wang, X., Zhu, K., Li, A., ... & Song, B. (2022). Three-nucleotide periodicity of nucleotide diversity in a population enables the identification of open reading frames. Briefings in Bioinformatics, 23(4), bbac210.

[16] Zhong, S., & Feng, J. (2022). CircPrimer 2.0: a software for annotating circRNAs and predicting translation potential of circRNAs. BMC bioinformatics, 23(1), 215.

[17] Djebali, Sarah, Carrie A. Davis, Angelika Merkel, Alex Dobin, Timo Lassmann, Ali Mortazavi, Andrea Tanzer et al. "Landscape of transcription in human cells." Nature 489, no. 7414 (2012): 101-108.

[18]Hartford, C. C. R., & Lal, A. (2020). When long noncoding becomes protein coding. Molecular and cellular biology, 40(6), e00528-19.

[19]Sieber, Patricia, Matthias Platzer, and Stefan Schuster. "The definition of open reading frame revisited." Trends in Genetics 34, no. 3 (2018): 167-170.

[20] Andrews, Shea J., and Joseph A. Rothnagel. "Emerging evidence for functional peptides encoded by short open reading frames." Nature Reviews Genetics 15, no. 3 (2014): 193-204.

[21] Schlesinger, Dörte, and Simon J. Elsässer. "Revisiting sORFs: overcoming challenges to identify and characterize functional microproteins." The FEBS Journal 289, no. 1 (2022): 53-74.

[22] Basrai, Munira A., Philip Hieter, and Jef D. Boeke. "Small open reading frames: beautiful needles in the haystack." Genome research 7, no. 8 (1997): 768-771.

[23] Chu, Qian, Jiao Ma, and Alan Saghatelian. "Identification and characterization of sORF-encoded polypeptides." Critical reviews in biochemistry and molecular biology 50, no. 2 (2015): 134-141.

[24] Yao, R. J. R., Andrade, J. G., Deyell, M. W., Jackson, H., McAlister, F. A., & Hawkins, N. M. (2019). Sensitivity, specificity, positive and negative predictive values of identifying atrial fibrillation using administrative data: a systematic review and meta-analysis. Clinical epidemiology, 11, 753.

[25] Chu, Qian, Jiao Ma, and Alan Saghatelian. "Identification and characterization of sORF-encoded polypeptides." Critical reviews in biochemistry and molecular biology 50, no. 2 (2015): 134-141.

[26] Birney, E., Andrews, T. D., Bevan, P., Caccamo, M., Chen, Y., Clarke, L., ... & Clamp, M. (2004). An overview of Ensembl. Genome research, 14(5), 925-928.

[27] Fu, L., Niu, B., Zhu, Z., Wu, S., & Li, W. (2012). CD-HIT: accelerated for clustering the next-generation sequencing data. Bioinformatics, 28(23), 3150-3152.

[28] Hsu, P. Y., & Benfey, P. N. (2018). Small but mighty: functional peptides encoded by small ORFs in plants. Proteomics, 18(10), 1700038.

[29] Ingolia, N. T., Ghaemmaghami, S., Newman, J. R., & Weissman, J. S. (2009). Genome-wide analysis in vivo of translation with nucleotide resolution using ribosome profiling. science, 324(5924), 218-223.

[30] Weiss, R. B., & Atkins, J. F. (2011). Translation goes global. Science, 334(6062), 1509-1510.

[31] Olexiouk, Volodimir, and Gerben Menschaert. "Using the sORFs. Org database." Current protocols in bioinformatics 65, no. 1 (2019): e68.

[32] Leong, A. Z. X., Lee, P. Y., Mohtar, M. A., Syafruddin, S. E., Pung, Y. F., & Low, T. Y. (2022). Short open reading frames (sORFs) and microproteins: an update on their identification and validation measures. Journal of Biomedical Science, 29(1), 19.

[33] Sagi, O., & Rokach, L. (2018). Ensemble learning: A survey. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, 8(4), e1249.

[34] NCBI Resource Coordinators. "Database resources of the national center for biotechnology information." Nucleic acids research 44, no. D1 (2016): D7-D19.

[35] Hao, Y., Zhang, L., Niu, Y., Cai, T., Luo, J., He, S., ... & Chen, R. (2018). SmProt: a database of small proteins encoded by annotated coding and non-coding RNA loci. Briefings in Bioinformatics, 19(4), 636-643.

[36] Zhao, Yi, Hui Li, Shuangsang Fang, Yue Kang, Wei Wu, Yajing Hao, Ziyang Li et al. "NONCODE 2016: an informative and valuable data source of long non-coding RNAs." Nucleic acids research 44, no. D1 (2016): D203-D208.