Improving Protein Interaction Prediction in GenPPi: A Novel Interaction Sampling Approach Preserving Network Topology

Alisson Silva, Carlos Marquez, Iury Godoy, Lucas Silva,
Matheus Prado, Murilo Beppler, Natanael Avila, and Anderson Santos *

April 3, 2025

Abstract

Computational prediction of protein-protein interactions (PPIs) is crucial for understanding cell biology and drug development, offering a promising alternative to costly and time-consuming experimental methods. While the original GenPPi software represented an advance in *ab initio* PPI network prediction from bacterial genomes, its reliance on high sequence similarity limited it. In this work, we introduce the GenPPi 1.5 and enhance prediction capabilities. We incorporated a Random Forest (RF) algorithm, trained on 60 biophysical features derived from amino acid propensity indices, to overcome the previous limitation by classifying protein similarity even in low sequence identity scenarios (targeting $> 65\%$ identity). Furthermore, to address the computational complexity arising from the increased number of potential interactions generated by the RF model, especially within extensive conserved phylogenetic profiles, we developed and integrated the Reduced Interaction Sampling (RIS) algorithm. RIS stochastically samples interactions within these extensive profiles, optimizing performance for complete genome analysis. Our multifaceted methodology included integrating RF for efficient similarity classification and implementing RIS to reduce the computational burden. We evaluated the effectiveness through extensive simulations across different configurations (operational modes, Top_N selected nodes, number of analyzed genomes). We rigorously assessed the network stability and structural integrity using topological metrics and statistical tests. Results demonstrate that RF integration significantly broadens GenPPi's predictive power. Application to the *Buchnera aphidicola* genome revealed an overlap of up to 62% with interactions documented in the STRING database, validating prediction accuracy. Analysis of the RIS algorithm showed that while introducing some randomness, critical node identification remains robust, particularly for Top_N values ≥ 100, indicating that the edge reduction does not significantly compromise network integrity. In conclusion, combining Machine Learning and RIS represents a significant advancement. GenPPi 1.5 overcomes the high-similarity dependency while efficiently handling complex genomes, providing a robust and scalable alignment-free solution for PPI prediction. GenPPi is available at `https://genppi.facom.ufu.br/` and on GitHub: `https://github.com/santosardr/genppi` and allows users to train custom models for specific genomic contexts.

Keywords: Protein, Interaction, Networks, Random Forest, alignment-free, RIS algorithm, Computational Biology

*All authors are with the Faculty of Computing (FACOM), Federal University of Uberlândia (UFU), Uberlândia, MG, Brazil. Corresponding author: Anderson Santos (e-mail: santosardr@ufu.br).

1 Introduction

Protein-protein interactions (PPIs) orchestrate fundamental cellular processes, making their identification crucial for understanding biological systems and developing therapeutic strategies [1]. While experimental methods provide valuable data, they are often costly and time-consuming [2], motivating the development of computational prediction approaches [1]. However, a significant challenge arises when analyzing the vast number of newly sequenced or poorly annotated genomes, mainly from bacteria and archaea. Many computational methods rely heavily on existing experimental data, structural information, or extensive functional annotations (e.g., [3, 4]), limiting their applicability precisely where discovery potential is high - in less-characterized organisms [5].

Addressing this gap requires robust *ab initio* methods capable of predicting PPI networks directly from genomic sequences with minimal reliance on external databases beyond sequence repositories. The GenPPi software was initially developed with this goal, utilizing genomic context features like conserved gene neighborhoods, phylogenetic profiles, and gene fusion events. We developed the program to analyze genomes represented in multiple files of protein sequences. [5]. While representing an advance, the original GenPPi (v1.0) faced a critical limitation: its protein similarity assessment required high sequence identity ($> 90\%$), significantly restricting its sensitivity and potentially missing biologically relevant interactions between more divergent proteins. Furthermore, simply lowering this threshold introduces a secondary challenge: the combinatorial explosion of potential interactions, overwhelming downstream analyses.

This work presents GenPPi 1.5, designed to overcome this dual challenge by integrating a sensitive Random Forest (RF) model for similarity assessment with the novel Reduced Interaction Sampling (RIS) algorithm to manage computational complexity.

Computational approaches to PPI prediction encompass a diverse range of methodologies [1]. Structure-based methods leverage 3D protein coordinates for docking or interface analysis but require known or accurately predicted structures, which are often unavailable [3] [6]. Other methods rely on integrating diverse high-throughput data like gene coexpression or protein complex copurification, limiting their use to well-studied organisms with extensive experimental datasets [4]. Machine learning, particularly deep learning applied to sequence data, has shown great promise but typically requires large, well-annotated training sets specific to particular organisms or interaction types [6]. Algorithms designed to predict protein-protein interactions can use gene ontology [2] or combine it with other strategies [7]. However, we showed that more than gene ontology is needed to produce satisfactory results in the prediction process [8]. We extensively reviewed the literature considering recent years, and no state-of-the-art software can predict interaction networks using exclusively genomic data efficiently.

The main contributions of this work are thus: (i) An alignment-free protein similarity prediction method integrated into GenPPi, utilizing a Random Forest classifier trained on biophysical features to effectively identify similarities below conventional sequence identity thresholds. (ii) The novel Reduced Interaction Sampling (RIS) algorithm is designed to manage the computational complexity arising from increased interaction candidates in sensitive prediction modes, thereby enabling scalable analysis. (iii) Rigorous validation of the machine learning model's performance across diverse bacterial genomes and a comprehensive evaluation of the RIS algorithm's impact on network topology preservation. (iv) An enhanced, parallelized, and open-source GenPPi 1.5 tool that significantly advances the state-of-the-art for practical *ab initio* PPI network prediction, particularly for understudied genomes. Moreover, the possibility of crafting a training model empowers researchers to optimize GenPPi's performance for their specific organisms of interest, potentially achieving even greater accuracy and relevance in their PPI predictions.

This paper elaborates on these contributions, presenting the methods employed for model train-

ing and RIS development (Section 2, evaluating their performance and impact through extensive testing (Section 3), and discussing the results in the context of current PPI prediction challenges (Section 4)."

1.1 State of the Art

In the context of predictions of protein interactions, the STRING (Search Tool for the Retrieval of Interacting Genes/Proteins) stands out as one of the leading state-of-the-art tools. STRING is a database that brings together, in an integrated way, known and predicted protein interactions, covering multiple sources of evidence, such as biochemical experiments, genetic associations, and gene coexpression. Its applicability extends to more than 2000 organisms, allowing comparative analyses between species. The platform offers an intuitive interface for visualizing interaction networks, facilitating the identification of functional modules and the interpretation of complex biological contexts. The constant updating and inclusion of new functionalities make STRING a valuable tool for bioinformatics and systems biology researchers, significantly advancing knowledge about protein interactions and their implications in health and disease [9].

Other tools are available for predicting protein-protein interaction networks, but not all are applicable or comparable to GenPPi in all contexts. For example, some of these tools address specific organisms, do not allow data upload for analysis, or are limited to data from multicellular organisms (eukaryotes). These limitations make comparing these tools directly with GenPPi, especially concerning unpublished, little-known, or studied complete genomes, impractical. GenPPi, on the other hand, allows the prediction of interaction networks from any genome of bacteria or archaea. The flexibility in creating interaction networks distinguishes GenPPi from other methodologies and makes it particularly useful for researchers working with genomes that other tools may not cover. Therefore, while being aware of the other tools and methodologies available is helpful, it is crucial to understand the limitations that prevent a direct comparison with GenPPi.

For example, in [7], which uses information from known complexes, gene expression profiles, Gene Ontology terms, and subcellular localization information for the discovery of essential proteins, is a theoretical algorithm and does not have an associated software for direct implementation in the code of the GenPPi.

Regarding [10], which proposes a new method to predict unknown protein-protein interactions, it is important to clarify that the associated software presented functionality problems despite its theoretically interesting proposal. Attempts to execute the code made available by the authors were unsuccessful, and our requests for assistance still need to be answered. Therefore, it was not possible to compare our results with theirs.

In addition, during the development of GenPPi, it was demonstrated that the interaction network accuracy produced using Gene Ontology was inferior to an *ab initio* generation [8].

The topological analysis of protein-protein interaction networks (PPI) is a relevant aspect of bioinformatics and systems biology, serving as one of the primary motivations for developing version 1.5 of GenPPi. A study entitled *Topological scoring of protein interaction networks* [11] reinforces the need for tools such as GenPPi, which facilitate topological analysis in PPI networks, contributing to a better understanding of protein interactions and their functional implications.

2 Materials and Methods

2.1 Features

The source code named *Features.lisp* is a recent addition to GenPPi in version 1.5, an in-house program that uses a file called *propensity.dat* as a basis for mapping specific features of proteins. This file contains specific values for each amino acid, allowing the program to create a propensity index for each characteristic of each amino acid present in all proteins in a multi-fast file. The data in the *propensity.dat* file represent different physicochemical characteristics of the amino acids, such as acidity/basicity (BASIC, ACID), polarity (POLAR, NONPOLAR), mass (MASS, MASSMR) and other specific indices (PARJ860101, JOND750101, EISD840101, JURD980101), in a total of ten characteristics available in the database of the AAindex website [12].

Each row of the *propensity.dat* file represents a characteristic, and each column is an amino acid. The numerical values represent the propensity of each amino acid for this characteristic. These indices were selected based on their effectiveness in classifying proteins according to the most common subcellular sites in prokaryotic organisms, such as the membrane, cytoplasm, cell wall surface, and secreted proteins [13].

The program reads the FASTA file, which contains several amino acid sequences, and the propensity file, which contains information about the physicochemical properties of amino acids. For each sequence in the FASTA file, the *Features* program calculates a histogram that counts the occurrence of each amino acid in the sequence. In addition, it calculates the weighted sum of the physicochemical properties for the entire sequence and the start, end, and middle regions of a sequence.

For each trait, the *Features* program calculates the value along the entire protein. Also, it segments the protein into the beginning, middle, and end, resulting in four values for each trait. This results in a subtotal of forty values per protein. In addition, simply counting the amount of each amino acid adds twenty more numerical characteristics to each protein, totaling sixty values defining a protein's propensity index. A more significant number of discriminative features allows for a more granular analysis of protein properties, as different regions of a protein can have different amino acid compositions and, therefore, different physicochemical properties.

2.2 Random Forest

The insertion of Random Forest as a machine learning algorithm in GenPPi emerges as a solution to improve the ability to predict protein-protein interaction networks (PPI) in bacterial genomes, a limitation faced by GenPPi 1.0 caused a non-optimal sampling of similar proteins. The challenge faced was to identify interactions between proteins with levels below 90%. Exact alignment or heuristic algorithms, such as blast, become computationally costly, especially in complex pangenomes, which is why we have sought alternative ways to identify protein similarity.

The choice of the Random Forest algorithm as a solution to the problem is justified by its ability to efficiently handle complex and high-dimensional datasets, as is the case with the genomic and proteomic information involved in the GenPPi [14]. Random Forest builds a collection of decision trees, each trained on a random subset of the available data and variables. This approach promotes diversity among trees, preventing dominant traits from exerting an excessive influence on the model and contributing to better generalization. In this work, we aimed to find a balance, improving sensitivity for lower-similarity pairs without generating an intractably large number of candidates for the downstream genomic analyses. Each tree forms a structure similar to a flowchart, contributing to joint decision-making [15].

We based the choice of the *cl-random-forest* library for incorporation into GenPPi on the need for robust and efficient implementation of Random Forest in Common Lisp [16]. The cl-random-forest library is an implementation of the Random Forest algorithm for SBCL. This library supports multiclass classification and univariate regression and is notable for its efficiency and accuracy, outperforming other popular implementations such as scikit-learn (Python/Cython) and ranger (R/C++). In addition, *cl-random-forest* includes a global refinement of Random Forest, as described in [17], making it even faster and more accurate than the standard implementation. This choice also facilitated integration within GenPPi's existing Common Lisp codebase.

The code used to train the Random Forest model used in GenPPi 1.5. is available in the GitHub repository `https://github.com/santosardr/non-CSPs/tree/main/src/lisp`. This file depends on the *cl-random-forest* and *cl-store* libraries installed via Quicklisp. Different training and testing sets were used with this interface to generate the final model for predicting similarities between protein pairs used in this new version of GenPPi.

2.3 Database

We compared the results obtained by the GenPPi software with the STRING data. STRING is recognized as a gold standard in protein interaction analysis, providing information on known interactions and predictions of interactions between proteins. For this purpose, we used the PostgreSQL relational database management system.

The database used for this analysis is available in the GenPPi repository on GitHub, under the name *compareba.dump*, since the study organism for comparison is *Buchnera aphidicola*. We designed this database for genome-related analyses *Ba_ak*, used as an example in this study. It consists of two main tables: *protein* and *related*. In Table *protein*, we stored the identifiers of the proteins and from which genome they originate, obtained from both the STRING interaction network and the GenPPi interaction network. The *related* table stores the interactions between proteins of the same genome, allowing the comparison of interactions between proteins.

We developed the *count_intersect_distinct* function to analyze protein interactions accurately. This function returns a table with the name of the genome and the number of unique interactions performed, considering the interactions with and without repetitions. In the context of STRING, where the interaction between A and B is considered different from that between B and A, the count of interactions without repetitions equals half the total interaction count. On the other hand, in the GenPPi, repeated interactions are removed through the GEPHI program, commonly used in this work to visualize the interaction networks generated by the GenPPi [18].

2.4 Experiments

2.4.1 Genomes Training and Testing

We turned to the National Center for Biotechnology Information (NCBI) due to its vast database and the availability of genomes of model pathogenic bacteria. We used the genomes of several species, including *Aeromonas hydrophila, Bacillus anthracis, Bacteroides fragilis, Borrelia burgdorferi, Clostridium botulinum, Clostridium tetani, Corynebacterium Jeikeium, Enterobacter huaxiensis, Enterococcus faecalis, Escherichia coli, Francisella tularensis, Helicobacter pylori, Moraxella Catarrhalis, Morganella morganii, Mycobacterium avium, Mycobacterium tuberculosis, Providencia rettgeri, Pseudomonas aeruginosa, Salmonella enterica, Staphylococcus aureus, Streptococcus pyogenes, Vibrio cholerae and Yersinia pestis*. We based the selection of these species on their relevance as pathogenic agents and on their extensive study in the scientific literature.

Data from model organisms, such as those found in the NCBI, can provide a solid basis for future investigations, allowing the confirmation of hypotheses and conclusions reached in our work. In addition, the availability of these genomes facilitates the creation of representative datasets and the performance of comparative analyses, contributing to the robustness and validity of the results obtained in our experiments.

2.4.2 Generating the Machine Learning Model

In the creation of the training file, we selected proteins from nine medically important genomes: *Bacillus anthracis, Clostridium tetani, Enterococcus faecalis, Escherichia coli, Francisella tularensis, Mycobacterium tuberculosis, Salmonella enterica, Staphylococcus aureus, and Vibrio cholerae.* The *src/tools/makearff.bash* script performs a series of manipulations on text files containing organic information, applying filters and converting them to the ARFF format, which is essential for training the algorithm. We used *src/tools/makearff.bash* to compare the nine genomes. This process generated a file with proteins identified with their characteristics and the class to which they belong (positive or negative).

The original full training dataset exhibited a significant imbalance between positive (8,858) and negative (318,309) protein pairs. Recognizing that such class imbalance can negatively impact classifier performance, we compared the performance of the Random Forest classifier using the default parameters provided by the `cl-random-forest` library against a set of empirically chosen custom parameters. This comparison was conducted across seven training subsets with varying negative-to-positive instance ratios (from 1:1 to 7:1), all evaluated on the independent *Corynebacterium* test set (Table 1).

Table 1: Performance Comparison of Random Forest Models Trained with Standard vs. Custom Hyperparameters across Different Negative:Positive Instance Ratios, Evaluated on the Independent *Corynebacterium* Test Set.

Training Set (Approx. Ratio) (Neg:Pos)	**Standard Parameters**			**Custom Parameters**		
	Sensitivity (Test Set)	**Specificity (Test Set)**	**F1-Score (Test Set)**	**Sensitivity (Test Set)**	**Specificity (Test Set)**	**F1-Score (Test Set)**
`1` (1:1)	0.4803	0.8712	0.5160	0.9988	0.3868	0.5238
`2` (2:1)	0.2342	0.9762	0.3590	0.9534	0.7235	0.6879
`3` **(3:1)**	0.2055	0.9814	0.3261	**0.8829**	**0.8466**	**0.7556**
`4` (4:1)	0.1649	0.9851	0.2727	0.6428	0.9370	0.7028
`5` (5:1)	0.1243	0.9895	0.2151	0.5651	0.9471	0.6565
`6` (6:1)	0.1075	0.9923	0.1903	0.5197	0.9536	0.6273
`7` (7:1)	0.0980	0.9948	0.1760	0.4659	0.9576	0.5856

Test set (*Corynebacterium*): 2477 Negative, 837 Positive instances. Custom Parameters: :n-tree 500, :max-depth 30, :n-trial 10. Standard parameters are the defaults provided by the cl-random-forest library.

The results in Table 1 highlight a clear trade-off. The standard parameters consistently yielded very high Specificity, particularly as the negative:positive ratio increased, but suffered from drastically reduced Sensitivity (Recall), rendering them unsuitable for GenPPi's discovery-oriented goals as they would miss most true similarities at higher, more realistic imbalance ratios.

Conversely, the custom parameters demonstrated significantly higher Sensitivity across all tested ratios. Given that GenPPi's primary goal is not only to confirm known interactions but also to facilitate the discovery of potentially novel ones - which often depend on identifying less obvious similarities missed by high-stringency methods - maximizing Sensitivity without excessively compromising Specificity was the primary selection criterion. The F1-score for the custom parameters

peaked at the 3:1 ratio in this evaluation, offering a strong balance between Sensitivity (0.8829) and Specificity (0.8466). While the 1:1 and 2:1 ratios offered even higher sensitivity, their specificity was considerably lower, leading to potentially unmanageable false positives. The subsequent filtering steps within GenPPi, which rely on analyzing genomic context, are designed to handle a moderate level of false positives. However, starting with a reasonable specificity is still beneficial. Therefore, we chose the 3:1 ratio trained with custom parameters for the final model. It provides the best input for the overall process by maximizing the capture of true potential similarities for downstream validation while maintaining acceptable specificity, representing the optimal compromise for GenPPi's objectives.

The custom hyperparameters used for the final model, justified by this comparative analysis showing their advantage over library defaults for our specific task, were configured as follows. The final model utilized 500 decision trees (`:n-tree 500`). To prevent overfitting while allowing sufficient complexity, the maximum depth of each tree (`:max-depth`) was programmatically set to half the number of input features plus one (resulting in a depth of 30 for our 60 features). The number of random feature subsets evaluated at each node split (`:n-trial`) was increased to 10 from the default of 1 to improve split quality potentially. We used default values for bagging ratio (`:bagging-ratio 1.0`) and minimum samples per leaf node (`:min-region-samples 1`).

A key strength of GenPPi lies in its flexibility regarding the underlying machine learning model. While we provide a pre-trained model optimized using the 3:1 ratio and the custom parameters described above, users are not limited to this default. The GenPPi architecture allows for creating custom models tailored to specific taxonomic groups or research questions. To facilitate this, we provide the maker script (`src/tools/makearff.bash`, available on the GenPPi GitHub repository), which converts protein sequence data in FASTA format into the ARFF format required for training the Random Forest classifier. Users can then follow the instructions provided in the repository (`src/tools/craftting-RF-training-model.lisp`) to generate a binary model file (`model.dat`) using any training data and desired parameters. Users should know the need to install the library `cl-random-forest` before crafting their training models.

We can call the program with the following command: *./randomforest similar_test.arff xxxx similar_training.arff*, where the first parameter is the name of the program, the second parameter *(similar_test.arff)* is the test file containing the target genome proteins, the third parameter *(xxxx)* indicates the number of negative proteins present in the test file, in digit format. It is important to note that we carefully structured the ARFF file so that all negative proteins precede positive ones. This ordering is critical, as we use true labels for each data instance in training. The Random Forest program, implemented in Lisp, uses this information to calculate the model's accuracy under construction, accounting for successes and errors. The last parameter represents the training file. Like the test file, the training file follows the pattern of negative instances preceding positive ones. This compliance allows training and test files to be interchangeable, providing flexibility in machine learning model evaluation and validation.

2.4.3 Using the GenPPi-trained model

The GenPPi initially employed the simple counting of the occurrences of the twenty-six amino acids (including six non-standardized) as a method of analysis. However, in the enhanced version, incorporated by the sixty features generated by the *Features* program, in the default mode of execution, called *Features* mode, we consider two proteins similar if they present at least twenty-five of these characteristics with a maximum difference of one or two units. This last number is a value obtained empirically. Changing this number affects the ability to generate inferences of correct similarities. If this number were increased too much, for example, close to the sixty features

that GenPPi 1.5 uses, GenPPi 1.5 in *Features* mode would pass like GenPPi 1.0 when it used twenty-six amino acid possibilities.

We employed the same sixty features used by the *Features* algorithm in the *cl-random-forest* program, which performs machine learning to create more complex networks of proteins. We can trigger the GenPPi machine learning mode through the optional *-ml* parameter. To use the machine learning mode, the user must download the previously trained Random Forest model to recognize similar protein pairs with more than 65% identity and size alignments as determined by BLASTP [19].

In the new version of GenPPi 1.5, we standardized some parameters to simplify the user experience. The *-ppcomplete* parameter, used to perform PPI predictions by the conserved phylogenetic profiling method without applying filters, is now enabled by default; the program will make the predictions without reducing the number of predicted interactions.

In addition, the type of neighborhood expansion of a recurrent gene controlled by the *-expt* parameter is now *fixed*. The size of the main window *(-w1)* and the required amount of genes conserved within window 1 *(-cw1)*, which were set to ten and four, respectively, are now also default values. Therefore, without extra parameters, GenPPi 1.5 will run with these default settings, simplifying the process and allowing users to focus more on interpreting the generated results.

Once one clones our example repository, it is possible to perform tests on the *Buchnera Aphidicola* genome. The program can be run with the following command (in the Linux operating system): *./genppi -dir test1*. The first parameter represents the version of the GenPPi program selected for the machine, with multiple builds available for different memory usage limits. The second parameter *('-dir')* indicates the name of the directory where we stored the genomes in the form of multifasta files (one file per genome), and the last parameter *('test1')* is the name of the directory in question.

For instance, we can fire the program's machine learning mode (in Linux) with the command: *./genppi -dir test1 -ml*. The parameters are the same, except the last one *('-ml')*, which instructs the program to use machine learning to identify similar proteins.

2.5 RIS (Reduced Interaction Sampling) Algorithm

When employing machine learning for PPI prediction in GenPPi, a significant increase in identified interactions was observed compared to approaches without machine learning. This increase, often driven by large conserved phylogenetic profiles involving hundreds of genes, represents an additional challenge for the subsequent topological analysis of the resulting network due to combinatorial explosion. For instance, a profile with $x = 500$ conserved genes could potentially generate $f(x) = \frac{(x-1)(x-2)}{2} = 124,750$ edges, rapidly leading to computationally intractable networks. This computational bottleneck underscores why simply maximizing the recall of any similarity classifier was insufficient; a reduction strategy became essential.

To mitigate this complexity while retaining the sensitivity gained from machine learning, the RIS (Reduced Interaction Sampling) algorithm, previously referred to as 'split,' was developed and incorporated into the GenPPi 1.5 source code (Algorithm 1). The core idea is to sample interactions from large phylogenetic profiles rather than generating all possible pairs, thereby reducing the edge density without drastically altering the network's core structure. RIS operates through three main steps:

1. **Shuffle:** The list of proteins within a large phylogenetic profile is randomly shuffled to eliminate any ordering bias.

2. **Subdivide:** The shuffled list is divided into multiple smaller sublists. We typically set the number of sublists to 90% of the original profile size.

3. **Independent Processing:** Each sublist is treated as an independent, smaller phylogenetic profile, and we generate interactions only *within* each sublist. A mechanism also ensures some overlap by randomly distributing several proteins among the sublists to maintain connectivity.

This strategy significantly reduces the number of pairwise comparisons and resulting edges, making downstream network analysis feasible even for complex pangenomes. The crucial question in the following sections is whether this sampling process preserves the network's essential topological features.

2.6 RIS Evaluation Methodology

To rigorously evaluate the impact of RIS on network topology, we conducted extensive computational experiments using the *Buchnera aphidicola* (Ba_ak) genome as a reference. We generated 50 independent network replicates for each of GenPPi's operational modes: *Features* (relying on biochemical characteristics) and *Machine Learning* (incorporating the Random Forest classifier). This replication strategy aimed to account for the stochasticity introduced by RIS's sampling process.

The experimental design involved varying key parameters to assess their influence on network stability:

- **Operational Mode**: Features vs. Machine Learning.
- **Number of Input Files (`NrFiles`)**: Ranging from 10 to 50, simulating different dataset sizes.
- **Top_N Nodes**: Selecting the top N nodes based on centrality, ranging from 10 (approx. 1.8% of the proteome) to 200 (approx. 35.8%), to evaluate RIS's effect at different levels of network granularity.
- **Topological Metrics**: Focusing on Degree, Betweenness Centrality, and Bridging Centrality as criteria for selecting Top_N nodes.

Our experimental design resulted in 180 distinct experimental configurations. For each generated network (in '.dot' format), we extracted topological metrics using Gephi [18] and saved them into '.csv' files for comparative analysis.

To quantify the stability and structural preservation under RIS, we employed a suite of statistical metrics comparing the replicates within each configuration:

- **Topological Metrics**: Degree (identifying hubs), Betweenness Centrality (identifying bottlenecks), and Bridging Centrality (identifying connectors between communities) [20]. We used these as criteria for Top_N selection and as measures of structural change.
- **Weighted Global Presence Mean (MGP)**: A custom metric developed to measure the consistent presence of critical nodes across replicates, weighted by their frequency of appearance, mitigating bias from sparsely appearing nodes.
- **Jaccard Coefficient**: To quantify the similarity between the sets of Top_N nodes across pairs of replicates [21].

- **Fleiss' Kappa (Adapted)**: To assess the multi-rater agreement (concordance) among all replicates on the selection of Top_N nodes, correcting for chance agreement [22].
- **Kendall Tau Rank Correlation Coefficient**: To measure the consistency of the hierarchical ranking of the Top_N nodes between pairs of replicates [23]. We specifically focused on the Kendall Ratio Positive (proportion of concordant pairs).
- **Kolmogorov-Smirnov (KS) Test (Multi-Sample)**: To compare the empirical distribution functions (CDFs) of the chosen topological metric (e.g., Degree) across all replicates within a configuration, assessing distributional similarity [24].

These metrics collectively allowed for a multifaceted assessment of whether RIS maintains the predicted PPI networks' global and local structural properties, hierarchical organization, and distributional characteristics.

2.7 Parallelism

The first GenPPi version was monoprocessed. We implemented parallelism in three code bottlenecks. The first critical point was the call of the function *call-similar-test* to determine a high similarity between protein pairs, the function now responsible for calling our Machine Learning implementation. We can find the second crucial point in the functions prefixed by *phylogenetic-profiles* responsible for inferring a conserved PP under different conditions triggered by specific parameters. We can find the last point at the functions *execute-expansion-fixed* and *execute-expansion-dynamic* performing the conserved neighborhood checking among proteins. The GenPPi will use all the processors available in a computer but two, allowing for better machine control.

Algorithm 1 Pseudocode for the RIS Algorithm

```

Procedure replace-item(target, item, replacements)
    pos ← position of item in target
if pos then
        target ← concatenation of (subsequence of target up to pos),
            replacements, (subsequence of target from pos+1)
end if

Procedure substitute-node(target, number, node)
    header, replacements ← empty
if target not empty and node not empty and number > 0 then
   if number < length of target then
            header ← 5th element of the number-th element of target
      for each subnode in node do
                Add (header, subnode, length of subnode) to replacements list
      end for
   else
            replace-item(target, number-th element of target, replacements)
   end if
end if

Procedure shuffle-list(target)
    copy ← copy of target
for i from length of copy down to 2 do
            Swap (random(i)-th element of copy) with ((i-1)-th element of copy)
end for
return  copy

Procedure RIS-list(protein-list, dividend, redundants)
    protein-list-shuffled ← shuffle-list(protein-list)
    n ← length of protein-list-shuffled
    sublist-size, sublists, pivotal-sublist, non-pivotal-sublists ← empty
if dividend < n and redundants < n then
            sublist-size ← round(n / dividend)
            sublists ← subdivision of protein-list-shuffled into
                sublists of size sublist-size
            dividend ← length of sublists
end if
for each pivotal index from 0 to dividend-1 do
            pivotal-sublist ← pivotal-th element of sublists
            non-pivotal-sublists ← sublists without pivotal-sublist
   for each redundant do
      if there are elements to choose from non-pivotal-sublists then
                    source_sublist_index ← random index of non-pivotal-sublists
                    source_sublist ← non-pivotal-sublists[source_sublist_index]
                    element_index ← random index of source_sublist
                    element ← source_sublist[element_index]
         if the element is not in the pivotal-sublist then
                            add element to pivotal-sublist
                            remove element from source_sublist
         end if
      end if
   end for
            update the pivotal-th element of sublists to pivotal-sublist
end for
return  sublists
```

When we analyze two or more genomes whose organisms are evolutionarily close, the tendency is for the number of genes in conserved phylogenetic profiles to increase significantly. For example, it is very common to have phylogenetic profiles conserved with more than five hundred genes in organisms with more than two thousand genes. Therefore, the number of edges initially created would be 124,251 edges, with each gene having 499 edges connected to the others. For another with 700 conserved genes, we would have 243,951 edges, with each gene having 699 edges. With few large profiles conserved in a genome, it is very common that at the end of the generation of the interaction network, we have edges that exceed half a million or even a million. The final amount of edges is proportional to the level of similarity between the genomes.

To slow down this rapid growth in the number of edges and average degrees of the graph's vertices, we thought of a process that could make a meaningful sampling of the interactions existing in phylogenetic profiles with several hundred conserved genes. Sampling aims to create as many interactions as possible between the genes with many genes without necessarily including all the edges. In this way, we were able to show that gene A interacts with gene B without necessarily including all the interactions of genes A and B. We call this sampling approach RIS, alluding to how the approach works by separating a profile's large set of genes into many smaller portions. The RIS generates several smaller profiles equal to 90% of the size of the original profile but places in each derived profile a smaller number of randomly selected genes.

Initially, the *RIS* promotes a disorder of the list of similar proteins known as *shuffle*. Later, we subdivided this list into several sublists equal to 90% of the number of elements in the original list. We randomly assign each sublist about half minus one element of the undivided unordered list. The program is then run separately on each sub-list, reducing the number of possible combinations and, consequently, the total number of edges in the network. Splitting and executing separately into sub-groups effectively reduces the number of interactions, facilitating topological analysis of the resulting network.

When considering this strategy, a relevant question is the fate of proteins that do not form edges. Although we miss some interactions, the tests carried out showed that after the application of the *RIS*, the result of the network of interactions practically did not change in terms of analyses of the topological importance of nodes, that is, in addition to reducing the number of edges in the network, which facilitates the analysis of the network of protein-protein interactions (PPI), The quality of the underlying data was not compromised, maintaining node differentiation in centrality measures. The explanation is that the decrease in the degree of the nodes was done homogeneously due to randomness, preserving the ranking of the most connected nodes. This approach provides an effective solution to address the increased number of interactions resulting from machine learning while maintaining the quality and integrity of the data for subsequent analysis by the PPI network.

The application of these methods yielded the results presented below.

3 Results

3.1 Performance Analysis of the Random Forest Classification Model

Table 2 presents the calculated performance metrics for the final Random Forest classification model, trained with the 3:1 negative-to-positive ratio and custom hyperparameters (as determined in Section 2.4.2), across the 26 diverse genome test sets. We created the test data for each genome or pair by identifying protein pairs with $> 65\%$ sequence identity via BLASTP (positive class) versus those below this threshold (negative class), using the methodology described earlier.

The evaluation across these independent test sets reveals the performance profile of the optimized model. The average Specificity is at approximately 90.0%, while the average Sensitivity is

Table 2: Random Forest Classification Performance Metrics on Various Genome Test Sets (Model Trained with 3:1 Ratio and Custom Hyperparameters).

Genomes / Test Set	**Confusion Matrix Counts**				**Performance Metrics**		
	TN	**FN**	**TP**	**FP**	**Sensitivity**	**Specificity**	**F1-Score**
Aeromonas hydrophila	1713	14	26	35	0.6500	0.9799	0.5149
Aeromonas hydrophila - Bacteroides fragilis	3548	112	84	273	0.4286	0.9286	0.3038
Aeromonas hydrophila - Clostridium botulinum	3347	61	68	93	0.5271	0.9730	0.4690
Aeromonas hydrophila - Enterobacter huaxiensis	4277	39	69	112	0.6389	0.9745	0.4775
Bacteroides fragilis	1835	98	58	238	0.3718	0.8852	0.2566
Bacteroides fragilis - Clostridium botulinum	3469	145	100	296	0.4082	0.9214	0.3120
Clostridium botulinum	1634	47	42	58	0.4719	0.9657	0.4444
Enterobacter huaxiensis	2564	25	43	77	0.6324	0.9708	0.4574
Enterobacter huaxiensis - Clostridium botulinum	4198	72	85	135	0.5414	0.9688	0.4509
Enterobacter huaxiensis - Bacteroides fragilis	4399	123	101	315	0.4509	0.9332	0.3156
Morganella morganii	938	18	95	91	0.8407	0.9116	0.6355
Providencia rettgeri	1066	62	207	125	0.7695	0.8950	0.6889
Yersinia pestis	1625	1461	3768	51	0.7206	0.9696	0.8329
Streptococcus pyogenes	320	9	13	7	0.5909	0.9786	0.6190
Mycobacterium avium	2463	164	266	493	0.6186	0.8332	0.4474
Mycobacterium avium - Corynebacterium jeikeium	2843	270	392	543	0.5921	0.8396	0.4909
Mycobacterium avium - Moraxella catarrhalis	2593	173	285	506	0.6223	0.8367	0.4564
Corynebacterium jeikeium	380	106	126	50	0.5431	0.8837	0.6176
Corynebacterium jeikeium - Moraxella catarrhalis	510	115	145	63	0.5577	0.8901	0.6197
Moraxella catarrhalis	130	9	19	13	0.6786	0.9091	0.6333
Helicobacter pylori	171	21	21	6	0.5000	0.9661	0.6087
Helicobacter pylori - Borrelia burgdorferi	284	331	449	73	0.5756	0.7955	0.6897
Helicobacter pylori - Pseudomonas aeruginosa	3570	81	82	175	0.5031	0.9533	0.3905
Pseudomonas aeruginosa	3399	60	61	169	0.5041	0.9526	0.3476
Pseudomonas aeruginosa - Borrelia burgdorferi	3512	370	489	236	0.5693	0.9370	0.6174
Borrelia burgdorferi	113	310	428	67	0.5799	0.6278	0.6942
Mean	-				**0.5734**	**0.8998**	**0.5136**

57.3%. Consequently, the average F1-Score, which balances precision and recall (Sensitivity), is approximately 51.4% (Table 2, last row).

This performance profile, characterized by high Specificity and moderate Sensitivity, warrants careful interpretation within the context of the overall GenPPi pipeline. While moderate sensitivity might seem disadvantageous in isolation, high specificity is crucial for GenPPi's operational efficiency. The downstream analyses within GenPPi operate on the pairs initially flagged as similar by the RF model. Given that the potential search space for pairs is vast (O(N^2) for N proteins), minimizing False Positives (FP) through high Specificity is paramount. A model with significantly lower specificity, even if slightly more sensitive, would generate an unmanageably large number of false leads, rendering the subsequent computationally intensive genomic context analyses intractable.

Furthermore, maximizing sensitivity in the initial similarity screen is balanced against the need for manageable downstream processing. GenPPi's core goal is to infer interaction networks based on conserved *groups* of genes. Identifying a sufficient number of correct similarity links *within* such groups, enabled by the model's moderate sensitivity (0.5734 average), is often enough to flag the entire set of proteins for interaction inference based on genomic evidence. Discovering *every* possible similar pair might be redundant.

Significant variability in performance across different test sets is still observed (Table 2). For example, the model achieves high Sensitivity and F1-Score for organisms like *Morganella morganii* (Sens=0.841, F1=0.635) and *Providencia rettgeri* (Sens=0.770, F1=0.689), and particularly high precision for *Yersinia pestis* contributing to a strong F1-score despite moderate sensitivity (Sens=0.721, F1=0.833). Conversely, sensitivity and F1-score remain lower for pairs involving more distant organisms or those with potentially unique protein characteristics, such as *Pseudomonas aeruginosa* pairs (e.g., Sens=0.504, F1=0.348 for intra-genome test) or some inter-genome tests involving *Bacteroides fragilis* (e.g., Sens=0.408, F1=0.312 vs *C. botulinum*), while still generally maintaining high Specificity. The variability of RF results highlights the inherent challenges in creating a universally performing similarity model based solely on these features. It underscores the importance of evaluating performance across diverse genomic backgrounds. It is important to note that users can further optimize performance by training custom models using their datasets, as described in Subsection 2.4.2.

Therefore, the trained model strikes a balance, prioritizing confidently rejecting non-similar pairs (high Specificity) while capturing a significant portion of potentially similar pairs (moderate Sensitivity). The resulting F1-Score reflects this balance, deemed optimal for GenPPi's specific architecture, where the initial RF classification acts as a critical filter to provide a tractable, high-confidence set of candidate pairs for the more biologically informative downstream genomic analyses. The objective is not merely to replicate BLAST results at a lower threshold but to efficiently identify plausible candidates for interaction based on combined evidence.

3.2 RIS Evaluation Results

The comprehensive evaluation across 180 configurations yielded extensive data, publicly available as detailed in [25]. Here, we summarize the key findings regarding the impact of the RIS algorithm.

Features Mode: In this mode, the RIS algorithm demonstrated negligible impact on network topology. Across nearly all configurations (85 out of 90), the Weighted Global Presence Mean (MGP) was 100%, with the lowest value observed being 99.76%. Similarly, Jaccard coefficients were consistently near 1.0, and Fleiss' Kappa indicated perfect agreement. These metrics outcomes demonstrate that when using the deterministic Features mode, RIS effectively reduces computational load without compromising the structural stability or reproducibility of the network predictions.

Machine Learning Mode: As anticipated, introducing RIS in the Machine Learning mode led to greater variability than the Features mode, primarily due to the stochastic nature of both the ML predictions and the RIS sampling. However, the analysis revealed clear trends related to the `Top_N` parameter:

- **Impact of `Top_N` on Stability:** The `Top_N` parameter emerged as the most critical factor influencing network stability. Increasing `Top_N` consistently led to higher stability across all metrics. For instance, MGP values (averaged across metrics and `FilesNum`) increased substantially, ranging from approximately 13-33% for `Top_N`=10 to 70-76% for `Top_N`=200 (Fig. 1). The results from Fig. 1 indicate that analyzing a larger fraction of the network's core structure mitigates the random effects introduced by RIS.

- **Preservation of Node Hierarchy:** The Kendall Tau correlation analysis showed remarkable stability in ranking critical nodes. For `Top_N` $\geq$ 100, the Kendall Ratio Positive approached or reached 1.0 across most configurations (Fig. 1), indicating preservation of the relative importance hierarchy of the main nodes despite the edge reduction performed by RIS.

- **Distributional Similarity:** The multi-sample KS test distance generally decreased as `Top_N` increased (Fig. 1), suggesting that the overall degree distributions become more consistent across replicates when considering a larger network core.

- **Comparison of Topological Metrics:** While all metrics showed improved stability with larger `Top_N`, Betweenness Centrality often exhibited slightly higher MGP and Fleiss' Kappa values compared to Degree or Bridging Centrality, suggesting it might be marginally more robust in identifying consistently critical nodes under RIS sampling. Heatmaps visualizing pairwise Jaccard coefficients further illustrated the increasing consistency with larger `Top_N` (Fig. 2).

Impact of `FilesNum`: Varying the number of input files showed less consistent effects. While Fleiss' Kappa generally increased with more files (suggesting better identification of a common core structure), MGP showed a slight decrease, potentially due to increased noise or variability introduced by analyzing more diverse datasets simultaneously. Jaccard and Kendall Tau correlations exhibited more erratic behavior relative to the number of files.

In conclusion, the evaluation demonstrates that the RIS algorithm is a viable and effective strategy for managing the computational complexity of PPI network generation in GenPPi, particularly when using the Machine Learning mode. While it introduces variability, especially for small `Top_N` values, the essential topological structure and hierarchical ranking of critical nodes are well-preserved when analyzing a sufficiently large portion of the network core (`Top_N` $\geq$ 100 recommended for a balance between computational efficiency and structural integrity). The stability in Features mode is near-perfect, validating its use without significant concern for stochastic effects. These findings support using RIS within GenPPi 1.5 as a practical approach to enable the analysis of large-scale genomic datasets.

3.3 Comparison of Results: GenPPi v1.0 vs. Use of Features and Machine Learning Features

The analysis of the results obtained by GenPPi in version 1.0 compared to incorporating the *Features* and *Machine Learning* functionalities in GenPPi 1.5 reveals a significant improvement in the prediction of protein interaction networks. We observed a consistent trend of increasing the

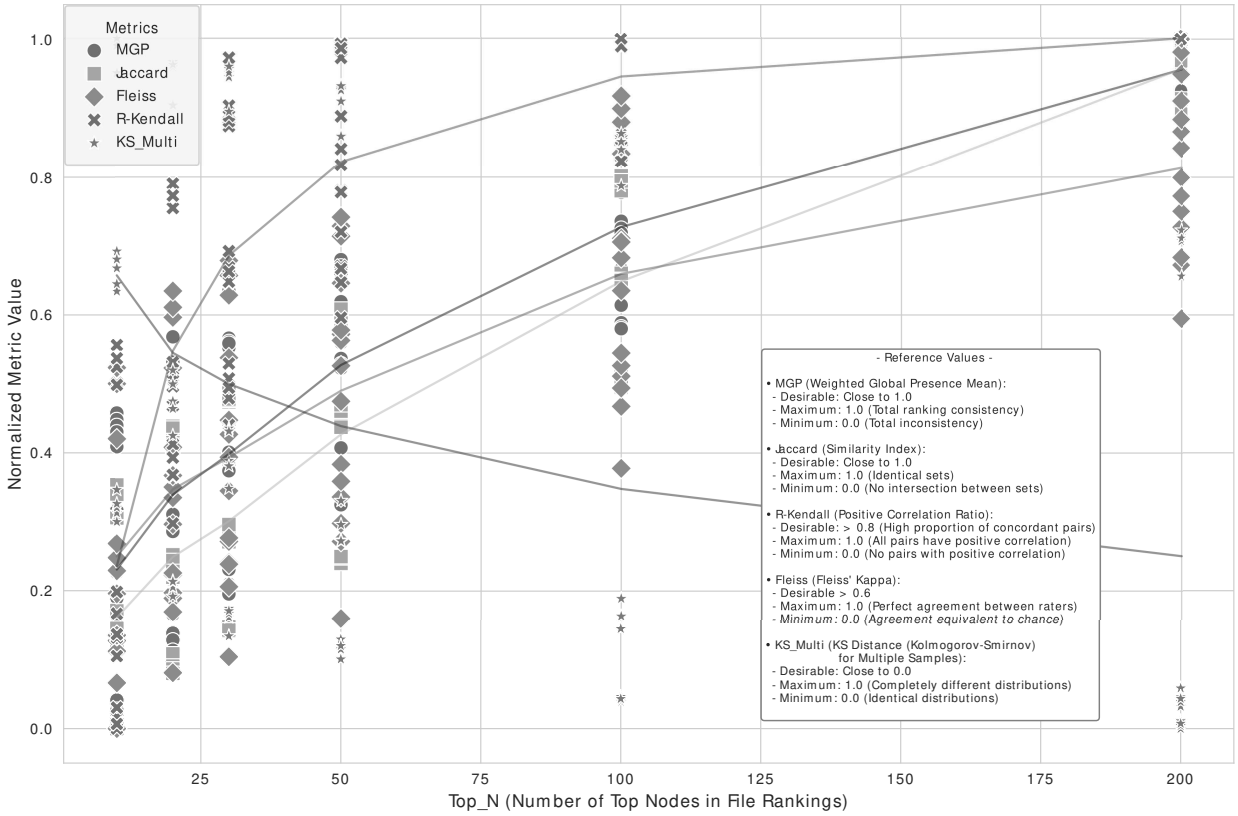

Figure 1: Behavior of Normalized Stability Metrics as a Function of Top_N in Machine Learning Mode. Metrics include Weighted Global Presence Mean (MGP), Jaccard Coefficient, Fleiss' Kappa, Kendall Ratio Positive (R-Kendall), and Multi-Sample KS Distance (KS_Multi, inverted scale: lower is better). All metrics to [0, 1] for comparison. Lines connect average values across replicates, showing increased stability (higher MGP, Jaccard, Fleiss, R-Kendall; lower KS_Multi) with larger Top_N values, indicating RIS preserves core network structure better when analyzing larger node sets.

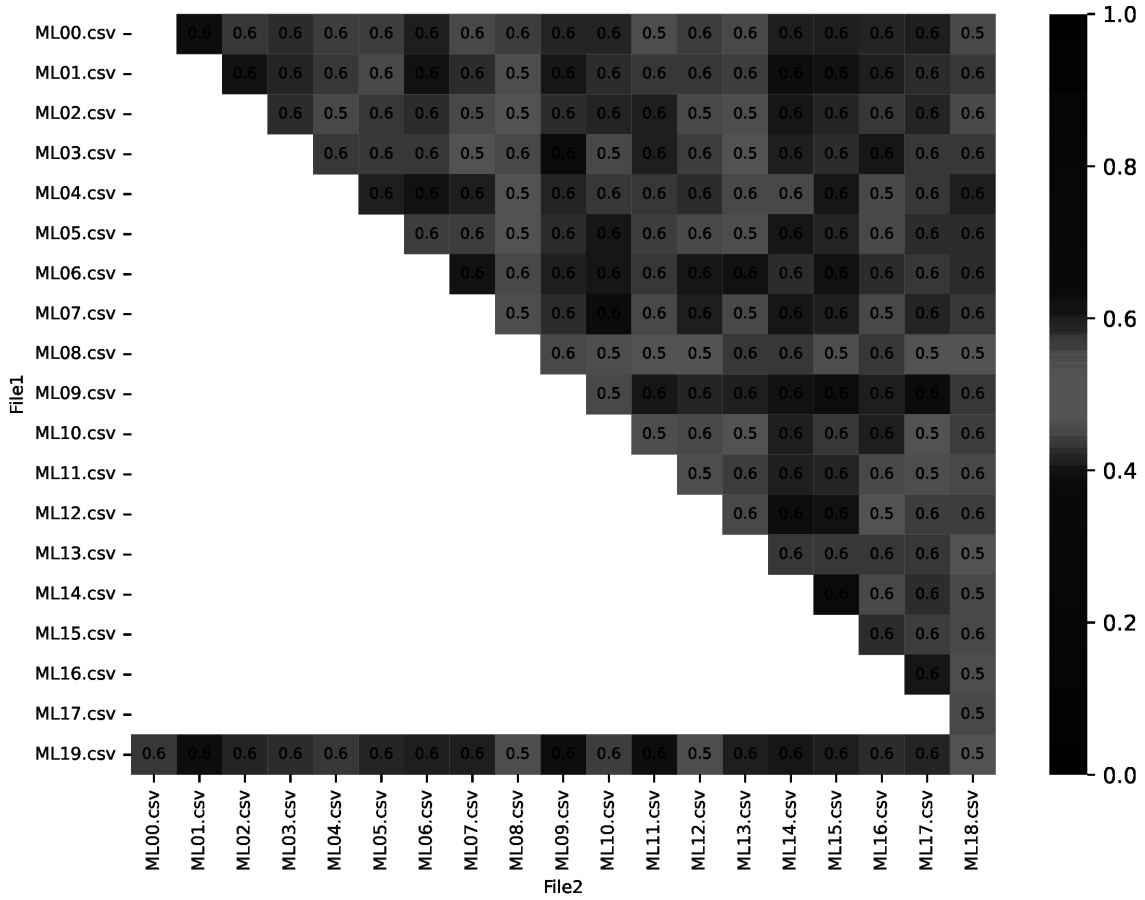

Figure 2: Example Heatmap of Pairwise Jaccard Coefficients for RIS Evaluation. This plot shows the similarity (Jaccard index) between the sets of Top_N nodes from different replicate runs (files ML00.csv to ML19.csv from a specific test case: ML mode, Top_N=200, 20 files, Degree metric). Axes represent pairs of replicate files. The grayscale intensity indicates the Jaccard coefficient value (0.0 to 1.0); in this scheme, darker shades represent both low (near 0.0) and high (near 1.0) values, while lighter values intermediate the scale (around 0.5) are lighter. Therefore, darker shades indicate high consistency between replicates representing high Jaccard indices (values closer to 1.0). Observing such heatmaps across different Top_N values revealed an increased prevalence of these darker shades (indicating higher consistency) for larger Top_N.

number of nodes and edges, indicating a greater complexity and scope of the networks generated after implementations.

Initially, in version 1.0 of the GenPPi, the genomes of *Buchnera aphidicola* resulted in a relatively low number of nodes and edges, as evidenced in the data in table 3. However, after introducing the *Features* and *Machine Learning* features, there was a notable increase in the number of nodes and edges in all the analyzed samples. This expansion of protein interaction networks suggests a greater ability of the improved software to identify and predict relationships between proteins, providing a more comprehensive and detailed view of the underlying biological processes.

Table 3: Comparison of Genppi Performance with Features and Machine Learning on Different Datasets

File	**Number of Proteins by Genomes**	**GenPPi 1**		**GenPPi 1.5**			
				Features		**Machine Learning**	
		Nodes	*Edges*	*Nodes*	*Edges*	*Nodes*	*Edges*
Ba_Bp.dot	553	46	91	217	1746	539	30220
Ba_Ua.dot	591	111	314	450	6115	581	40049
Ba_G002.dot	621	120	395	529	9166	617	41969
Ba_Sg.dot	620	88	203	430	5583	614	41483
Ba_Ak.dot	559	119	339	448	7917	559	44005
Summary	2944	484	1342	2074	30527	2910	197726

The results also indicate an upward trend in the complexity of the networks generated, reflected in the significant increase in the number of nodes and edges. In summary, the comparative results between the different versions of GenPPi demonstrate the benefits of using additional resources in predicting protein interaction networks.

3.4 Temporal Efficiency Analysis

To address the computational efficiency of the enhanced GenPPi (v1.5) relative to the original version (v1.0), comparative runtime tests were performed using the *Buchnera aphidicola* genome dataset on a 40-core processor machine, employing a dynamic neighborhood expansion window starting at 3 for all runs. GenPPi v1.0, using its simpler similarity heuristics and serial processing, completed the analysis rapidly in approximately 4 seconds, identifying 156 interactions just for the *Ba_ak* genome. GenPPi v1.5 in Features mode, which replaces the simple heuristic with a more complex comparison across 60 biophysical features, took significantly longer (39 seconds) despite utilizing parallel processing. This increase reflects both the higher intrinsic cost of the feature comparison and the larger workload imposed on downstream analyses by the substantially greater number of identified interactions (6,863). The Machine Learning mode of GenPPi v1.5, incorporating the Random Forest classifier and the RIS algorithm, required the longest execution time (117 seconds), yielding the highest number of interactions (45,934). We attributed this further increase to the computational expense of evaluating the 500-tree Random Forest model for similarity prediction and the subsequent processing of the vastly expanded set of potential interactions by the parallelized genomic context algorithms (even with RIS mitigating combinatorial explosion). These results quantify the expected trade-off: the enhanced sensitivity and broader interaction discovery capabilities of GenPPi 1.5, particularly in ML mode, come at the cost of increased computational time, even when leveraging parallel execution on multi-core hardware.

3.5 Comparison with STRING

The data presented below refer to the *Ba_ak* genome of the bacterium *Buchenera aphidicola* and were obtained through a SQL query to the *compareba.dump* database. The SQL file *count_intersect_distinct.sql* contains the definition of a function called *count_intersect*. This function returns a table with information about the intersection of different genomic datasets. Function code involves identifying unique elements and intersections between those elements and calculating metrics such as the proportion of unique intersections and interactions between elements. After defining the function, a SELECT statement calls the *count_intersect* function and returns the query results.

The conserved phylogenetic profile infers interactions for pairs of proteins that have identical profiles. Pairing proteins with similar profiles can also interact. To incorporate this notion of similarity rather than exact identity, the *-ppdifftolerated* parameter in GenPPi establishes the tolerated difference between profiles. The flexibility provided by the *-ppdifftolerated* parameter allows for a more comprehensive analysis of protein-protein interactions, considering the similarity of profiles [5].

Table 4 presents the results of different implementations of GenPPi, varying parameters such as the number of proteins in a phylogenetic profile and the tolerated difference between the profiles of a pair of proteins. Each row in Table 4 represents a specific implementation. The table features various metrics, including *query_unique, string_intersect_unique, unique_intersect_ratio, and string_proportion* for different parameters. For example, the term *tol* is synonymous with the *-ppdifftolerated parameter*. The N number accompanying the *RIS* represents the attempts to determine whether the *RIS* algorithm would be triggered when a profile set had more than N proteins. The standard that has been established (*mlris*) is N=100.

In table 4, we have results for the *Features* and the *Machine Learning* modes. Only the sixty traits described in section 2.1 were used with a cut-off factor of twenty-five identical features between a pair of proteins, considering them identical for eventual interaction conservation analyses. The term *ml* is synonymous with the *Machine Learning* mode of GenPPi 1.5. The *ml* mode is triggered when the *Features* mode concludes that, based on the lack of at least twenty-five identical features among a pair of proteins, a pair of proteins is not identical.

The *query_unique* column represents the number of implementation-related individual interactions. In comparison with STRING, column *string_intersect_unique* indicates the number of unique interactions that match those of STRING, while column *unique_intersect_ratio* represents the ratio of the number of unique interactions in GenPPi to STRING. The *string_proportion* column indicates the proportion of interactions in GenPPi that are also present in STRING.

Note duplicated values referring to the row corresponding to the STRING because the data source in the STRING considers the interaction between A and B different from B and A. In the other data, there is not this duplicity. Therefore, the actual value of *unique_intersect_ratio* and *string_proportion* in the line corresponding to the STRING is 1 and 1, instead of 0.5 and 2, respectively. The reason is that the *count_intersect* function is intended to obtain metrics for implementations made in GenPPi and not for STRING.

Unique_intersect_ratio is a crucial metric for understanding the effectiveness of different GenPPi implementations. This ratio represents the number of predicted interactions divided by the number of interactions intersecting with the STRING. Therefore, for every interaction that matches the STRING, many other interactions do not belong to the STRING. We can interpret the existence of interactions absent in STRING as an inverted distance measure of the STRING, where a *unique_intersect_ratio* of 1 would indicate a perfect match with the STRING. The higher the *unique_intersect_ratio* value, the greater the difference between the phylogenetic profiles of a pair of proteins is allowed to classify them as interacting, which may result in more predicted interactions

Table 4: Comparison of GenPPi 1.5 Implementations against STRING for *B. aphidicola* (Ba_ak).

Implementation	**Unique Pairs**	**STRING Overlap**	**Unique / Overlap Ratio**	**STRING Proportion**
string	27041	54082[a]	0.50[b]	2.00[b]
Features	7917	2773	2.86	0.10
mlris[c]	**49077**	**16681**	**2.94**	**0.62**
mlristol1[c,d]	103743	26924	3.85	1.00
mlristol2[c,d]	128579	29846	4.31	1.10
mlristol3[c,d]	144468	32278	4.48	1.19
mlris35tol3[c,d,e]	135617	23840	5.69	0.88
mlris35tol2[c,d,e]	114998	20034	5.74	0.74

[a] STRING overlap for itself counts directed pairs (A-B, B-A).
[b] Ratio and Proportion for STRING adjusted assuming 27,041 unique interactions.
[c] 'mlris' indicates Machine Learning mode with RIS algorithm (default N=100).
[d] 'tolN' indicates using the '-ppdifftolerated' parameter N (1, 2, or 3).
[e] 'mlrisN' indicates the RIS threshold set to N (here, N=35 is shown).
Column Definitions: **Unique Pairs**: Number of unique interaction pairs predicted by GenPPi (A-B/B-A counted once). **STRING Overlap**: Number of unique GenPPi pairs also present in STRING (unique interactions = 27,041). **Unique / Overlap Ratio**: Ratio of Unique Pairs to STRING Overlap; values > 1 indicate predictions beyond STRING. **STRING Proportion**: Recall against STRING (STRING Overlap / 27,041).

by GenPPi. However, some of these additional interactions may not be present in STRING. These interactions, absent from STRING but supported by GenPPi's analysis of conserved genomic features, represent putative candidates for discovery. Interactions predicted by GenPPi in STRING can be useful for discovering potential new interactions but can also lead to more false positives. Therefore, it is important to strike a balance when adjusting the *unique_intersect_ratio* value.

It is important to consider that a high average degree of edges, determined by the value of *unique_intersect_ratio*, can hinder a topological analysis, as it generates many interactions. Therefore, when evaluating the effectiveness of different GenPPi implementations, it is crucial to consider not only the STRING match but also the structure and topology of the resulting network.

As the value of *tol* increases (from *tol1* to *tol3*), the *unique_intersect_ratio* also increases. Greater values to *tol* suggest that allowing a greater difference between phylogenetic profiles may result in greater agreement with the STRING. Similarly, increasing the number of proteins in a profile (from *mlris10* to *mlris35*) also appears to increase *unique_intersect_ratio*. However, it is important to note that *unique_intersect_ratio* higher does not necessarily indicate a better prediction of protein-protein interactions. While greater agreement with the STRING may suggest greater accuracy, it is also possible that some valid interactions may be lost. Therefore, it is crucial to consider the balance between sensitivity (the ability to identify interactions correctly) and specificity (the ability to exclude non-interactions correctly) when evaluating the effectiveness of different GenPPi implementations.

In addition, the *string_proportion* represents the result of dividing the number of single-intersect interactions of the specific GenPPi implementation by the number of STRING single-intersecting interactions (i.e., 27041), a measure of the proportion of interactions in GenPPi that are also present in STRING. For the implementation with *Machine Learning* and the use of the *RIS* algorithm - *mlris*, it obtained a percentage of 62%. For *mlris35tol3* and *mlris35tol2*, a percentage of 88% and

74% respectively. For *mlristol1*, *mlristol2* and *mlristol3*, values higher than 100%, being respectively 100%, 110% and 119%. Values higher than 100% occur in certain cases where the adopted implementation recognizes the STRING values of interactions repeated in different scenarios.

Among the various implementations, the most significant is *mlris*, which uses *Machine Learning* and *RIS* algorithm and generates a total of 49,077 unique interactions, representing a proportion of 1.81 times more interactions than the 27,041 unique interactions present in STRING. Of this total, 16,681 interactions coincide with STRING, representing 62% of the interactions predicted in STRING. The ratio between the number of unique interactions of the implementation divided by the number of interactions that intersect with the STRING was 2.94, indicating that for each coincident interaction, there is 1.94 interaction that does not match the STRING. These quantitative results demonstrate the ability of the GenPPi program equipped with the RIS algorithm to generate predictions of protein interactions with a significant proportion of overlap with the interactions documented in the STRING while presenting a network that is not extremely dense.

The presented results demonstrate significant improvements in GenPPi's capabilities. However, they also highlight important trade-offs and require careful interpretation within the broader context of PPI prediction.

4 Discussion

4.1 Interpreting Predictions and Network Density

The protein interactions predicted by GenPPi that are not present in the STRING database do not necessarily imply the non-existence of these interactions or the incorrectness of the GenPPi software. The correctly implemented algorithms for conserved phylogenetic profile and gene neighborhood traits conferred between the GenPPi versions speak in favor of a high probability that these interactions occur. However, an overly dense protein network can make identifying significant differences in the network topology difficult, hindering deeper analyses. On the other hand, an overly sparse network is also not ideal. Therefore, the search for an intermediate-density network becomes a relevant objective.

4.2 Performance Comparison: Features vs. Machine Learning Modes

Comparing the operational modes reveals significant differences in Table 4. The Features mode, operating without machine learning, generates a considerably smaller number of unique interactions (‘query_unique‘). In contrast, the Machine Learning mode produces substantially larger networks. This difference impacts the comparison with STRING: while ML implementations achieve higher overlap (‘string_proportion‘ reaching up to 1.19, indicating the capture of nearly all STRING interactions and potentially identifying duplicates present in STRING itself), the Features mode shows lower overlap (e.g., 0.10).

However, the increased capture rate in ML mode brings together a much larger number of predicted interactions not found in STRING, as shown by the higher ‘unique_intersect_ratio‘ values (reaching up to 5.74). Consequently, the ML-generated networks exhibit significantly higher edge density due to many unique predictions (‘query_unique‘), which can complicate subsequent topological analyses compared to the sparser networks from the Features mode. A clear trade-off between sparse and dense nets: the sparser Features network may be more amenable to topological inference while potentially missing some known interactions (lower ‘string_proportion‘). Conversely, the denser ML network captures more known biology (higher ‘string_proportion‘) but poses greater

challenges for analysis due to its complexity and the higher ratio of novel-to-known predictions (higher ‘unique_intersect_ratio‘).

4.3 Validation Against STRING Database

It is crucial that the predicted protein networks significantly overlap with the protein interactions documented in the STRING database. String overlapping validates the forecasts' accuracy and strengthens the results' reliability. The results suggest that the interactions generated by GenPPi significantly overlap with those predicted by STRING, which is considered a gold standard. The interpretation of these data indicates that GenPPi can capture substantial interactions similar to those of STRING, thus validating the software's effectiveness and relevance in generating protein-protein interaction networks. This quantitative comparison provides important insights into the quality and accuracy of the interactions produced by GenPPi relative to the standard set by STRING.

4.4 Utilizing Interaction Confidence Scores

An interaction can also have a high probability of being true or indicate a lower probability of being a real interaction. To workaround the problem of the high number of interactions, some parameters can be adjusted to increase the confidence of the probability of an interaction. GenPPi and STRING provide information on known interactions and predictions of protein interactions. STRING, for example, assigns a confidence value or a confidence score to indicate the strength or probability of the interaction. This value is calculated based on various data sources, including experimental evidence, gene coexpression, and sequence similarity. We displayed the confidence value assigned by STRING and GenPPi as a number between 0 and 1.

4.5 Overall Performance and Non-Randomness of Predictions

The results obtained through using GenPPi demonstrate its usefulness in improving the quality and efficiency of investigations. A key step contributing to this achievement was the RF model demonstrating high average specificity (90%) but moderate average sensitivity (57.3%), resulting in an average F1-Score of 51%. Such a moderate F1-Score highlights the model's prioritization of minimizing false positives. Another important point to discuss is the number of edges presented by the GenPPi interaction networks. The average number of proteins in our set of five *Buchnera aphidicola* genomes is about 590. All possible combinations of protein pairs, valid or not as interacting, are about 348,100 per genome. Considering our best outcome in Table 4, genome *mlris* and *string_proportion* 0.62, among all possibilities of interactions, we gathered 92,137 or just 26% of the odds, considering the *Machine Learning* mode. When we configure GenPPi with *Features* mode, the slice possible protein pairs from the odds is only 2%. Even in the worst GenPPi configuration scenario, genome *mlristol3*, we obtained 0.41% of the odds. However, we obtained 100% of the STRING predictions, followed by the configuration genome *mlristol1* getting a lower slice from the odds of 29% of unique interactions. These numbers evidence that the GenPPi interaction networks are not pure by chance. Despite some configurations giving rise to a higher number of candidate interactions, we rely on a significant number of interactions predicted by other software and for trustable algorithms.

4.6 Rationale for Random Forest Selection

While a wide array of machine learning algorithms, such as XGBoost or KNN, could potentially be applied to identify protein homology, a comprehensive benchmark against alternatives was not the primary focus of this work, nor would it necessarily reflect the specific constraints and optimization goals inherent to the GenPPi pipeline. The main objective was to enhance GenPPi's ability to efficiently predict PPIs within its existing architecture, which heavily relies on the subsequent, computationally intensive analysis of conserved phylogenetic profiles and gene neighborhoods.

Although increasing the recall of potential protein similarities is desirable, naively maximizing the number of predicted similarities exacerbates the computational burden of these downstream steps. GenPPi's core innovation lies in identifying similarities and intelligently filtering and prioritizing them based on genomic evidence to generate a manageable and biologically meaningful network. Simply using a classifier with the highest possible recall could overwhelm this filtering process.

The implementation of the RIS algorithm directly addresses this challenge. It allows leveraging the sensitivity of Random Forest to identify potential interactions while mitigating the prohibitive computational cost associated with analyzing an excessive number of candidates. Preliminary experiments with alternative algorithms indicated that some achieved slightly higher initial recall. However, considering that our main goal was not a machine learning benchmark, we opted not to use those, avoiding dramatically modifying our source code.

Furthermore, Random Forest offered a practical advantage due to the availability of the robust `cl-random-forest` library implemented in Common Lisp, the same language used for GenPPi's development. A native Common Lisp software facilitated seamless integration and leveraged the strengths of the Lisp environment. GenPPi, combining Random Forest's balanced performance with the RIS reduction strategy, achieves prediction rates characterized by moderate specificity (90%) and moderate sensitivity (57%), sufficient to generate high-quality PPI networks within the practical constraints of its analytical pipeline, striking a necessary balance between sensitivity and computational feasibility.

4.7 Validation of Novel Interactions

Beyond the choice of classifier, a critical aspect concerns the validation of interactions predicted by GenPPi, particularly those not found in reference databases like STRING. A crucial aspect of PPI prediction involves discovering novel interactions not previously documented. However, the experimental validation of computationally predicted interactions presents significant challenges common to all large-scale prediction methods. It is worth noting here that reference databases like STRING, while invaluable, contain many predicted interactions lacking direct experimental validation. The number of novel interactions proposed by methods like GenPPi (or, indeed, by established databases like STRING) can be considerable. No high-throughput computational method exists to empirically validate them biologically *en masse*.

GenPPi and databases like STRING rely heavily on computational predictions, with only a fraction of interactions typically having direct experimental support. Therefore, protein interactions predicted solely by GenPPi, absent in STRING, should not be automatically dismissed as false positives. These represent putative interactions, supported by the genomic evidence rigorously analyzed by GenPPi - rules validated across GenPPi versions. They serve as prioritized candidates for future experimental investigation, potentially leading to new biological insights, particularly in less-studied organisms where databases like STRING may have limited coverage.

The goal of GenPPi is not necessarily to perfectly replicate existing databases but to leverage

genomic context to propose biologically plausible networks, including novel interactions that warrant further study. While striving for significant overlap with known interactions (as demonstrated by the 62% overlap with STRING) validates the overall approach, the novel predictions highlight the discovery potential inherent in *ab initio* genome analysis. Focusing solely on replicating known data would limit the tool's utility for exploring the unknown biology of diverse organisms. Comprehensive experimental validation remains a resource-intensive bottleneck for the entire field, lying beyond the scope of the computational prediction tool itself.

4.8 Open Source Availability

Using open-source code in all versions of GenPPi presents numerous advantages for scientific research. Access to open-source code provides benefits such as customization, control, cost reduction, collaboration, transparency, and reliability. However, it is important to note that scientific sources such as STRING are not open to the public, limiting access to the source code and compromising the results' transparency and reliability. GenPPi version 1.5 is available for download at the following address: `https://genppi.facom.ufu.br/`

4.9 User's rules

A relevant aspect of the GenPPI software is its ability to transfer to the end user the decision about how many and which genomes to use in constructing the protein interaction network. All the code was written in Common Lisp and compiled using SBCL, allowing the creation of complex protein networks predicted from tens or hundreds of genomes. GenPPi stands out for its versatility and design, which makes it compatible with various operating systems and hardware configurations. We optimized program versions, meeting different needs and usage scenarios. It is easy to access and use, requiring no specific expertise in computing or biology. We designed the software to facilitate discovering novel relationships between proteins and their associated biological processes.

5 Conclusion

Based on the results obtained with GenPPi, it is possible to infer that the software offers functionalities valuable for molecular biology and bioinformatics researchers. GenPPi's ability to generate protein interaction networks from specific, unpublished genomic data without depending on external databases and with validated predictive performance allows researchers to explore protein interactions in lesser-studied genomes or organisms with unique characteristics, thus broadening the scope of scientific investigations. While direct experimental validation of all novel predictions remains a field-wide challenge, GenPPi provides candidates prioritized by strong genomic evidence. The results of the latest version of GenPPi demonstrate a significant advance in protein interaction studies.

Acknowledgments

We want to express our gratitude to the Federal University of Uberlândia for the support and resources provided during the completion of this work.

References

[1] Tao Tang, Xiaocai Zhang, Yuansheng Liu, Hui Peng, Binshuang Zheng, Yanlin Yin, and Xiangxiang Zeng. Machine learning on protein-protein interaction prediction: models, challenges and trends. *Briefings in Bioinformatics*, 24:1–11, March 2023.

[2] Lun Hu, Xiaojuan Wang, Yu An Huang, Pengwei Hu, and Zhu Hong You. A Novel Network-Based Algorithm for Predicting Protein-Protein Interactions Using Gene Ontology. *Frontiers in Microbiology*, 12, August 2021.

[3] Sharon Sunny and PB Jayaraj. Protein–protein docking: Past, present, and future. *The protein journal*, 41(1):1–26, 2022.

[4] Jamie Snider, Max Kotlyar, Punit Saraon, Zhong Yao, Igor Jurisica, and Igor Stagljar. Fundamentals of protein interaction network mapping. *Molecular Systems Biology*, 11(12):848, 2015.

[5] William F. Anjos, Gabriel C. Lanes, Vasco A. Azevedo, and Anderson R. Santos. GENPPI: standalone software for creating protein interaction networks from genomes. *BMC Bioinformatics*, 22:1–26, December 2021.

[6] Farzan Soleymani, Eric Paquet, Herna Viktor, Wojtek Michalowski, and Davide Spinello. Protein-protein interaction prediction with deep learning: A comprehensive review. *Computational and Structural Biotechnology Journal*, 20:5316, January 2022.

[7] Wei Zhang, Xiaoli Xue, Chengwang Xie, Yuanyuan Li, Junhong Liu, Hailin Chen, and Guanghui Li. CEGSO: Boosting Essential Proteins Prediction by Integrating Protein Complex, Gene Expression, Gene Ontology, Subcellular Localization and Orthology Information. *Interdisciplinary sciences, computational life sciences*, 13:349–361, September 2021.

[8] G. S. Oliveira and A. R. Santos. Using the Gene Ontology tool to produce de novo protein-protein interaction networks with IS_A relationship. *Genetics and molecular research : GMR*, 15, December 2016.

[9] Damian Szklarczyk, Andrea Franceschini, Stefan Wyder, Kristoffer Forslund, Davide Heller, Jaime Huerta-Cepas, Milan Simonovic, Alexander Roth, Alberto Santos, Kalliopi P Tsafou, Michael Kuhn, Peer Bork, Lars J Jensen, and Christian von Mering. STRING v10: protein-protein interaction networks, integrated over the tree of life. *Nucleic acids research*, 43:447–452, January 2015.

[10] Yan Kang, Xinchao Wang, Cheng Xie, Huadong Zhang, and Wentao Xie. BBLN: A bilateral-branch learning network for unknown protein-protein interaction prediction. *Computers in Biology and Medicine*, 167:107588, December 2023.

[11] Mihaela E. Sardiu, Joshua M. Gilmore, Brad D. Groppe, Arnob Dutta, Laurence Florens, and Michael P. Washburn. Topological scoring of protein interaction networks. *Nature Communications 2019 10:1*, 10:1–14, March 2019.

[12] Shuichi Kawashima, Piotr Pokarowski, Maria Pokarowska, and Minoru Kanehisa. AAindex: amino acid index database. *Nucleic acids research*, 28(1):374–374, 2000.

[13] Jianhua Jia, Zi Liu, Xuan Xiao, Bingxiang Liu, and Kuo-Chen Chou. Identification of protein-protein binding sites by incorporating the physicochemical properties and stationary wavelet transforms into pseudo amino acid composition. *Journal of Biomolecular Structure and Dynamics*, 34(9):1946–1961, 2016.

[14] Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

[15] Jaime Lynn Speiser, Michael E. Miller, Janet Tooze, and Edward Ip. A comparison of random forest variable selection methods for classification prediction modeling. *Expert Systems with Applications*, 134:93–101, November 2019.

[16] Satoshi Imai. CL-random-forest: Random forest in Common Lisp. GitHub, August 2024.

[17] Shaoqing Ren, Xudong Cao, Yichen Wei, and Jian Sun. Global refinement of random forest. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 07-12-June-2015:723–730, October 2015.

[18] Mathieu Bastian and Sebastien Heymann. Gephi : An Open Source Software for Exploring and Manipulating Networks. *ICWSM*, 8:361–362, 2009.

[19] Christiam Camacho, George Coulouris, Vahram Avagyan, Ning Ma, Jason Papadopoulos, Kevin Bealer, and Thomas L Madden. BLAST+: architecture and applications. *BMC bioinformatics*, 10(1):1–9, 2009.

[20] Getulio Pereira, Preetam Ghosh, and Anderson Santos. A Bridging Centrality plugin for GEPHI and a case study for Mycobacterium tuberculosis H37Rv. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 18(6):2741–2746, 2021.

[21] Paul Jaccard. Étude comparative de la distribution florale dans une portion des Alpes et des Jura. *Bull Soc Vaudoise Sci Nat*, 37:547–579, 1901.

[22] Joseph L Fleiss. Measuring nominal scale agreement among many raters. *Psychological bulletin*, 76(5):378, 1971.

[23] Maurice G Kendall. A new measure of rank correlation. *Biometrika*, 30(1-2):81–93, 1938.

[24] Kolmogorov An. Sulla determinazione empirica di una legge didistribuzione. *Giorn Dell'inst Ital Degli Att*, 4:89–91, 1933.

[25] Alisson Silva, Carlos Marquez, Iury Godoy, Lucas Silva, Matheus Prado, Murilo Beppler, Natanael Avila, and Anderson Santos. GenPPI RIS Algorithm Evaluation Dataset, March 2024. Version 1.5.