GigaScience, 10, 2021, 1–14

https://doi.org/10.1093/gigascience/giab042
Technical Note

TECHNICAL NOTE

Mantis: flexible and consensus-driven genome annotation

Pedro Queirós [1,*], Francesco Delogu [1], Oskar Hickl [2], Patrick May [2,*] and Paul Wilmes [1,*]

[1]Systems Ecology, Luxembourg Centre for Systems Biomedicine, University of Luxembourg, 6 Avenue du Swing, 4367 Esch-sur-Alzette, Luxembourg and [2]Bioinformatics Core, Luxembourg Centre for Systems Biomedicine, University of Luxembourg, 6 Avenue du Swing, 4367 Esch-sur-Alzette, Luxembourg

*Correspondence address. Pedro Queirós, Systems Ecology, Luxembourg Centre for Systems Biomedicine, University of Luxembourg, Université du Luxembourg 2, Avenue de l'Université L-4365 Esch-sur-Alzette. E-mail: pdqueiros@gmail.com http://orcid.org/0000-0002-0831-4261; Paul Wilmes, Systems Ecology, Luxembourg Centre for Systems Biomedicine, University of Luxembourg, Université du Luxembourg 7, Avenue des Hauts Fourneaux L-4362 Esch-sur-Alzette. E-mail: paul.wilmes@uni.lu http://orcid.org/0000-0001-8698-3770

Abstract

Background: The rapid development of the (meta-)omics fields has produced an unprecedented amount of high-resolution and high-fidelity data. Through the use of these datasets we can infer the role of previously functionally unannotated proteins from single organisms and consortia. In this context, protein function annotation can be described as the identification of regions of interest (i.e., domains) in protein sequences and the assignment of biological functions. Despite the existence of numerous tools, challenges remain in terms of speed, flexibility, and reproducibility. In the big data era, it is also increasingly important to cease limiting our findings to a single reference, coalescing knowledge from different data sources, and thus overcoming some limitations in overly relying on computationally generated data from single sources.
Results: We implemented a protein annotation tool, Mantis, which uses database identifiers intersection and text mining to integrate knowledge from multiple reference data sources into a single consensus-driven output. Mantis is flexible, allowing for the customization of reference data and execution parameters, and is reproducible across different research goals and user environments. We implemented a depth-first search algorithm for domain-specific annotation, which significantly improved annotation performance compared to sequence-wide annotation. The parallelized implementation of Mantis results in short runtimes while also outputting high coverage and high-quality protein function annotations.
Conclusions: Mantis is a protein function annotation tool that produces high-quality consensus-driven protein annotations. It is easy to set up, customize, and use, scaling from single genomes to large metagenomes. Mantis is available under the MIT license at https://github.com/PedroMTQ/mantis.

Keywords: bioinformatics; consensus; homology; HMM; protein function annotation

Background

On a cellular scale, life is, in essence, the activity and the interaction of a plethora of different molecules, among which proteins are responsible for the vast majority of processes. A primary task in understanding how biology works is to resolve its actors properly (e.g., the proteins) and place them into context. The past decades have seen the development of the (meta-)omics fields, unlocking an unprecedented amount of data and deepening our understanding in several fields of biology [1, 2]. Alongside the evolution of the technologies and the increase in data volume, the identification of proteins transitioned from purely experimental techniques (e.g., chemical assays and spectroscopy)

Received: 4 November 2020; **Revised:** 22 March 2021; **Accepted:** 14 May 2021

toward computational-based sequence analysis thanks to the discovery of the relationship between conservation of proteins' functions and sequences [3]. Therefore, the current challenges are to make use of the vast number of protein sequences and annotations available and to link new protein sequences to the previously established knowledge. High-throughput methods, such as next-generation sequencing, are able to produce a large amount of data, which then need to be analysed and interpreted. One of the ways to make sense of these data is through protein function annotation (PFA), which is, in the context of this article, the identification of regions of interest (i.e., domains) in a sequence and assignment of biological function(s) to these regions. This strategy has proven effective in the study of single organisms, as well as consortia [4–9]. Function prediction is based on reference data, i.e., transferring the function from protein X to the unknown protein Y if they are highly similar [3]. Different approaches may be used, the most common being the comparison of an unknown protein sequence to reference data composed of well-studied and functionally annotated proteins (homology-based methods) [10–16]. Other methods may infer function through the use of machine learning [10, 17], protein networks [18, 19], protein structure [20], or genomics context-based techniques [21], but these are not covered in this article. For sequence alignment, BLAST [22] or Diamond [23] are commonly used, whereas, for hidden Markov models (HMM) profiles, HMMER [24] is most widely used. In PFA, these tools are often integrated into larger pipelines to provide enhanced output interpretability, workflow automation, and parallelization [14–16, 25]. Some PFA tools target specific taxa [26], while others are designed with large-scale omics analysis in mind [27–29]; indeed, each PFA tool is designed to cater to its niche research topic. While experimental validation remains the gold standard, PFA, despite its many shortcomings [30], is an increasingly valuable strategy that aims to tackle the progressively more difficult task of making sense of the large quantities of data being continuously generated.

The most common method of processing candidate annotations (i.e., sequences or HMM profiles that are highly similar to the query sequence) involves capturing only the most significant candidate ("best prediction only" [BPO] algorithm). This PFA approach works well for single-domain proteins, but multi-domain proteins may have multiple putative predictions [31–33], whose location in the sequence may or may not overlap. This selection criterion may potentially lead to missing annotations and is therefore not suitable in complex PFA scenarios. To tackle this problem, domain-specific PFA is necessary. A simple approach, previously discussed in Yeats et al. [31], would be to order the predictions by their significance and iteratively add the most significant one, as long as it does not overlap with the already added predictions (henceforth referred to as the "heuristic" algorithm). Owing to the biased selection of the first prediction, this algorithm does not guarantee an optimal solution (e.g., a protein sequence may have multiple similarly significant predictions). It has been previously shown that incorporating prediction significance and length may produce better results [34]. We implemented a depth-first search (DFS) algorithm that improves on the previous approaches.

The selection of reference HMMs is also critical because PFA will ultimately be based on the available reference data. Whilst using unspecific HMMs to annotate a taxonomically classified sample may result in a fair amount of true-positive (TP) results (correct annotations), depending on the confidence threshold used, it may also increase the rate of false-positive (FP) results (over-annotation, due to a less strict confidence threshold) or false-negative (FN) results (under-annotation, due to a more strict confidence threshold) [35]. Using taxon-specific HMMs (TSHMM) rather than unspecific HMMs should, in principle, provide better annotations on a taxonomically classified sample, a feature that is already integrated into some PFA tools such as eggNOG-mapper [15] and RAST [16]. In essence, TSHMM-based annotation limits the available search space, which may have positive and negative consequences. Because the search space is more specific, the annotations produced should be of higher quality; however, this higher specificity of the TSHMM could also lead to under-annotation (incomplete reference TSHMMs) or mis-annotations (low-quality reference TSHMM) [36]. This underlines the necessity to use specific (e.g., TSHMMs) and unspecific HMMs in a complementary manner. In this regard, the use of multiple sources of reference data remains a challenging aspect of PFA, and, with multiple high-quality reference data sources available, it is increasingly important to coalesce knowledge from different sources. While some PFA tools allow for the use of multiple reference data sources, either as a separate [25] or a unified [15, 37] database, it is still challenging to integrate multiple data sources dynamically.

When using reference data from multiple high-quality sources, the most common and straightforward approach is to consider the output from each reference data source independently (e.g., [25]). However, by doing so, we overlook that many sources can overlap and/or complement each other. Commonly this is compensated for via manual curation, which is feasible only for a limited number of annotations. An automated approach would be to assume only the most significant annotation source for any given sequence and disregard other sources; this may result in vast losses of potentially valid and complementary information (e.g., database identifiers). Because this is not desirable, the challenge is in both deciding which source(s) provide the best annotation as well as identifying complementary annotations. In the present context, complementary annotations can be defined as functional annotations that are functionally similar but originate from difference data sources; as such, while functionally similar, different data sources are likely to contain information that is absent in other data sources and vice versa. This unique functional information (i.e., database identifiers or functional descriptions) may prove essential in downstream data analysis. A straightforward approach to verify whether functional annotations are functionally similar is to check whether they share a database identifier (ID), e.g.,

(i) Function: "Responsible for glucose degradation"; IDs: K00844, EC:2.7.1.1, PF03727
(ii) Function: "Responsible for glucose degradation"; IDs: P52789, PF03727, IPR022673

We can observe that the annotations (i) and (ii) share the database ID PF03727, thus it can be concluded that these annotations are functionally similar. If we were only to select the first annotation, we would ignore potentially useful information (IDs P52789 and IPR022673). However, it may be the case that no IDs are shared between the different annotations, e.g.,

(i) Function: "Responsible for glucose degradation"; IDs: K00844, EC:2.7.1.1
(ii) Function: "Responsible for glucose degradation"; IDs: P52789, IPR022673

We can observe that even though the annotations (i) and (ii) no longer share an ID, they still have the same function "Responsible for glucose degradation." Humans can quickly surmise that these annotations are the same because they share the same function description. Should the descriptions be identical or very similar, a machine could achieve the same conclusion with relative ease. However, in our experience, these free-text functional descriptions are often moderately or heavily dissimilar [38, 39], with only a few keywords allowing us to ascertain that they are indeed the same. This then makes it more difficult to use multiple reference data sources. For example:

(i) Function: "Responsible for glucose degradation"; IDs: K00844, EC:2.7.1.1
(ii) Function: "Protein is an enzyme and it is responsible for the breakdown of glucose"; IDs: HXK2_HUMAN

In such a scenario, someone trained in a biology-related field can quickly identify the most important words ("degradation"/"breakdown" and "glucose") in both sentences and conclude that both annotations point to the same biological function. The challenge is now to enable a machine, deprived of any intellect and intuition, to eliminate confounders (ubiquitous words, e.g., "the"), identify keywords and their potential synonyms, and reach the same conclusion. A possible strategy is to use text mining, which is the process of exploring and analysing large amounts of unstructured text data aided by software, identifying potential concepts, patterns, topics, keywords, and other attributes in the data [40]. Text mining has been previously used with biological data [41–45], and even more specifically with regards to gene ontologies [46–51] and PFA [43]. However, to our knowledge, there is no tool for the dynamic generation of a consensus from multiple protein annotations. This article solves the problem of scaling the integration of different annotation sources, integrating a compact and flexible text-mining strategy. We implemented a 2-fold approach to build a consensus annotation, first by checking for any intersecting annotation IDs and second by evaluating how similar the free-text functional descriptions are. This approach attempts to address 3 relevant issues with PFA [35, 36, 52, 53]: over-annotation, under-annotation, and redundancy. Another challenge in PFA is the lack of flexibility of some tools, as these are often intrinsically connected to their in-house–generated reference data and therefore hard to customize. In contrast, we developed a tool that, while offering high-quality unspecific and specific HMMs, is independent of its reference data, thus being customizable and allowing dynamic integration of new data sources.

We hereby present Mantis, a Python-based PFA tool that overcomes the previously presented issues, producing high-quality annotations with the integration of multiple domains and multiple reference data sources. Mantis automatically downloads and compiles several high-quality reference data sources and efficiently uses the available hardware through parallelized execution. Mantis is independent of any of the default reference data, resulting in a versatile and reproducible tool that overcomes the challenge of high-throughput protein annotation coming from the many genome and metagenome sequencing projects.

Mantis

Mantis is available at https://github.com/PedroMTQ/mantis, and its workflow (see Fig. 1) consists of 6 main steps: (i) sample pre-processing, (ii) HMM profile-based homology search, (iii) intra-HMM hit processing, (iv) metadata integration, (v) inter-HMM hit processing, and (vi) consensus generation. For future reference, an instance when an HMM matches with a protein se-

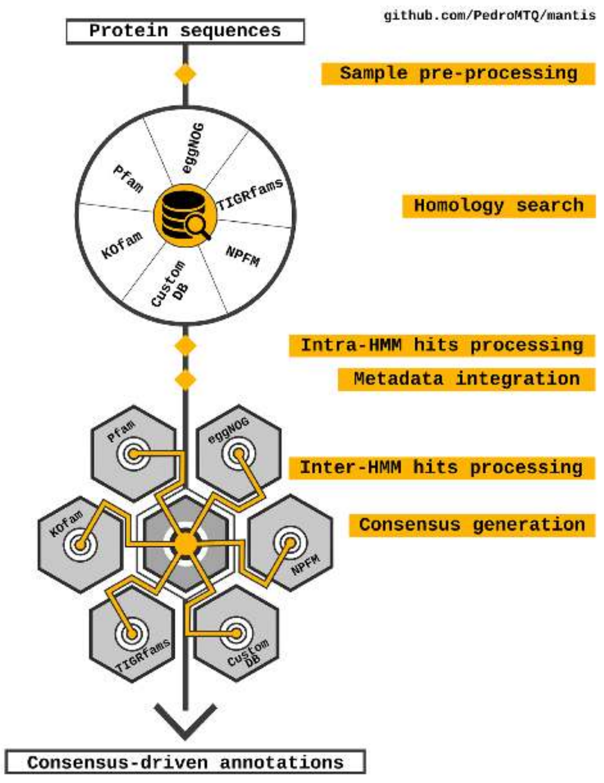

Figure 1: Overview of the Mantis workflow. KOfam [55], Pfam [56], eggNOG [57], NCBI protein family models (NPFM) [58], and TIGRfams [59] are the reference HMMs currently used in Mantis. CustomDB can be any HMM library provided by the user.

quence is referred to as a "hit." The workflow starts with sample pre-processing, in which the sample(s) is/are split into chunks. This is followed by homology search, where query sequences are searched against the available reference data using HMMER. During intra-HMM hit processing the DFS algorithm is used to generate and select the best combination of hits per HMM source; Fig. 2 shows how different algorithms may lead to a different selection of hits. Metadata integration adds the metadata (functional description and IDs) to the respective hits. During inter-HMM hit processing, the DFS algorithm is used to generate all the combinations of hits from all HMM sources (in this step all hits are pooled together). Finally, consensus generation ensures that the best combination of hits among all hits from the multiple reference data sources is selected. This combination is expanded by adding additional hits with consistent metadata (intersecting identifiers or similar functional descriptions) (see Methods section for a detailed description of all these steps). We provide default execution parameters; however, the user is free to fully customize Mantis, not only the parameters but also the reference databases used. Mantis requires a FASTA-formatted protein sequence file as input, where the user can also provide the organism's taxon to allow for taxon-specific annotation (TSA). Reference databases are downloaded automatically. The MANTIS.config file allows for configuration of the reference data and its respective weights and enables the compilation of specific eggNOG TSHMMs. For more details, see the documentation [54]. Owing to issues with Python's multiprocessing in MacOS, and the fact that HMMER is not available on Windows, Mantis is only available on Linux-based systems.

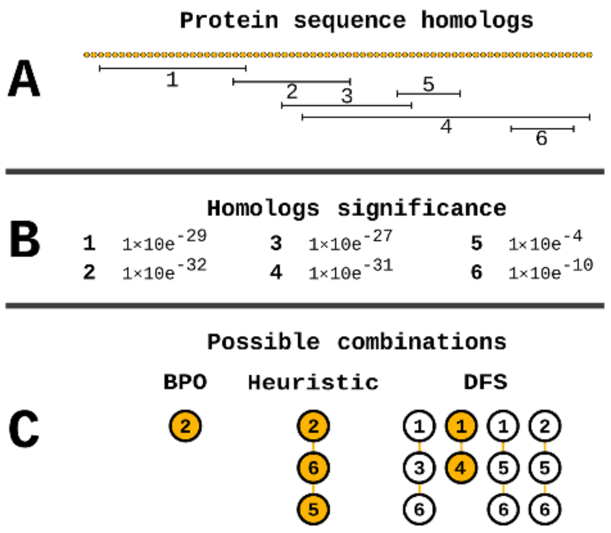

Figure 2: Homolog selection for the 3 hit-processing algorithms in Mantis. The selection of the hit(s) depends on the underlying algorithm. In the case of the portrayed protein with 6 hits (A) (which are overlapping to various degrees) that have varying significance values (B) the 3 algorithms would behave as follows: (i) BPO would select only the most significant hit (No. 2); (ii) the heuristic algorithm initially selects the most significant hit (No. 2), which then restricts (due to overlapping residues) the hits available for selection (hits 1, 3, and 4 can no longer be selected), leading to the selection of the next most significant hit (No. 6), and finally the selection of hit 5; (iii) the DFS algorithm generates all possible combinations of hits, which are then scored according to the e-value, hit coverage, and total combination coverage (for more details, see "Multiple hits per protein"). According to these parameters, the most likely combinations of hits would be hits 1 and 4.

Analysis

To analyse and validate the performance of Mantis, we performed several *in silico* experiments. We annotated a reference dataset containing curated protein entries from UniProt to set default parameters and evaluate the impact of different Mantis features: (i) impact of the e-value threshold, (ii) impact of the hit-processing algorithm, (iii) how each reference data source contributed to the final output, and (iv) impact of the consensus generation on annotation quality. Furthermore, we annotated several sequenced organisms, with and without TSHMMs, thus evaluating the impact of using taxon-resolved reference data. Finally, we compared Mantis against eggNOG-mapper [15] and Prokka [14]. A description of the samples used for this benchmark is available in "Sample selection." Prokka was only used for the annotation of prokaryotic data (i.e., all except for *Saccharomyces cerevisiae* and *Cryptococcus neoformans*). To compare the performance between the different tests, we calculated a confusion matrix for each test. For future reference, a TP occurs when a functional annotation (predicted from a PFA tool) shares ≥1 database ID with the respective reference annotation (e.g., Pfam ID), an FP when no database IDs are shared, an FN when the PFA tool does not annotate a protein sequence but a reference annotation is available, and a true-negative (TN) when the PFA tool does not annotate a protein sequence and no reference annotation is available. Precision is defined as TP/(TP + FP), recall as TP/(TP + FN), and F1 score (harmonic mean of precision and recall) as 2 × [(precision × recall)/(precision + recall)]. The F1 score is used as a performance metric. Further details on the benchmark are available in "Establishing a test environment."

Initial quality control

Function assignment e-value threshold

It is known that the e-value threshold directly affects annotation quality; however, no gold standard threshold exists [34]. Depending on the reference data source's size, quality, and specificity, we may use more or less stringent thresholds. It is therefore essential to test annotation quality with different thresholds. As such, we tested different static e-value thresholds and a dynamic threshold, which is described in "Testing different e-value thresholds." As can be seen in Supplementary Table 1, precision was similar across the range of e-value thresholds tested, with recall/sensitivity decreasing with lower e-value thresholds. Unexpectedly, unlike recall, precision was not directly correlated with the e-value threshold; indeed a maximum precision of 0.747 was obtained for the e-value threshold $1e^{-6}$, with precision slightly decreasing with more stringent e-value thresholds. A maximum F1 score of 0.827 was observed for the e-value threshold $1e^{-3}$; as such, we chose this value as the default e-value threshold for Mantis.

Impact of hit-processing algorithms

To understand whether the different hit-processing algorithms resulted in statistically significant differences in F1 scores, we created synthetic samples and performed pairwise comparisons between the DFS and the other algorithms: (i) DFS and heuristic and (ii) DFS and BPO. We rejected the H_0: "no differences in F1 score between the tested algorithms" in both comparisons because $P < 0.01$. The DFS algorithm resulted in a greater F1 score (mean = 0.827) than the heuristic (mean = 0.826) and BPO (mean = 0.816) algorithms. Further details on results can be found in Supplementary Table 2, and further details on the testing method can be found in "Testing hit-processing algorithms."

Impact of sample selection

Testing exclusively against well-annotated organisms is a recurring issue with protein annotation benchmarking, resulting in the re-annotation of sequences already present in the reference data used, leading to a biased annotation quality evaluation. To avoid this bias, we downloaded all the curated UniProt (i.e., Swiss-Prot) protein entries (as of 14 April 2020) and selected entries by their creation date such that we have 4 samples that contain protein entries created in different date ranges (2010–2020, 2015–2020, 2018–2020, and 2020). Samples with more recent protein entries are increasingly more likely to lack any proteins used to generate Mantis's reference data, which increases the likelihood that potential annotations are due to true sequence homology (and not to circular re-annotations). We annotated these samples using 3 different hit-processing algorithms (DFS, heuristic, and BPO), determining the impact of each on the F1 score.

As seen in Fig. 3, the F1 score decreased as the sample was restricted to more recent data. As seen in Supplementary Table 3, when comparing the hit-processing algorithms, we found that the DFS algorithm consistently outperformed the other algorithms, with an average F1 score 0.021 and 0.003 higher than the BPO and heuristic algorithms, respectively. In addition, the F1 score difference between the multiple hits algorithms (DFS and heuristic) and the single hit algorithm (BPO) increased as the entries in a sample were restricted to more recent years.

Contribution of the different reference data sources

We analysed each reference data source's contribution to the output annotation for the UniProt 2010–2020 sample. By check-

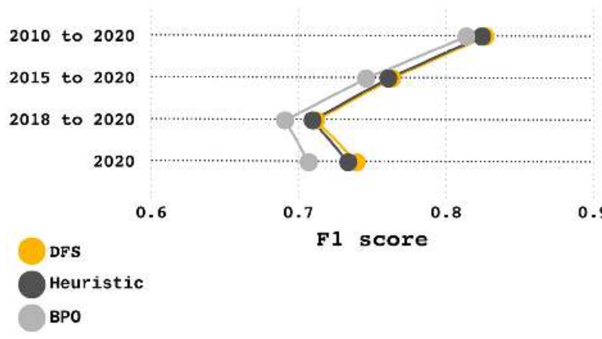

Figure 3: Annotation F1 score per hit-processing algorithm and sample. Overall, the DFS and heuristic algorithms achieve similar results, outperforming the BPO algorithm.

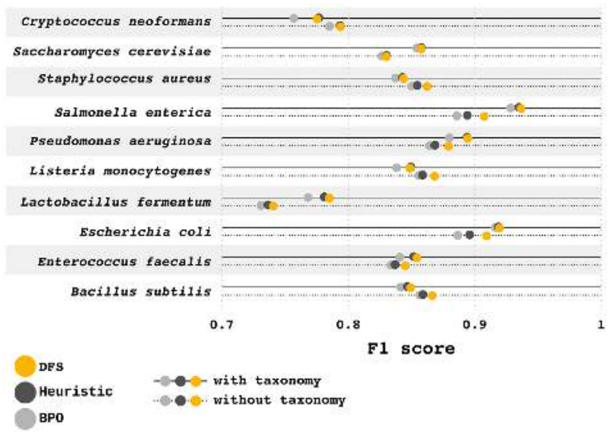

Figure 4: F1 score per hit-processing algorithm and organism, with and without using taxonomy information. F1 score was higher for well-studied organisms; TSHMMs also tend to perform better with these organisms.

ing the column "HMM_files" in the consensus_annotation.tsv file, we found that Pfam was present in 24.4% of the sequence annotations, KOfam in 62.37%, eggNOG in 76.52%, NPFM in 13.91%, and TIGRfam in 12.96%. Note that, because multiple reference data sources may be present in 1 sequence (due to the consensus generation and hit-processing algorithms), the sum of the previous values is >100%.

Impact of consensus generation

During consensus generation, 2 methods are used for checking the consistency of the hit metadata: ID intersection and text mining. We analysed the contribution of both methods for the annotation of the UniProt 2010–2020 sample and found that 35.10% of the consistency checks were due to the text-mining approach, and the remaining were due to ID intersection.

We also tested the impact of text mining on annotation performance: to do so, we annotated the Uniprot 2010–2020 sample but restricted the consensus generation in different manners and with different algorithms. Six different test conditions were created: (i) DFS with default consensus generation, (ii) DFS with consensus generation restricted to IDs (i.e., ID intersection but no text mining), (iii) DFS without consensus generation (i.e., neither ID intersection nor text mining), (iv) BPO with default consensus generation, (v) BPO with consensus generation restricted to IDs, and (vi) BPO without consensus generation. We also annotated the same sample using eggNOG-mapper—condition (vii). Prokka was not used here because the present sample contains non-prokaryotic data. The F1 scores were as follows: (i) 0.827, (ii) 0.790, (iii) 0.774, (iv) 0.814, (v) 0.779, (vi) 0.763, and (vii) 0.703. Further details can be found in Supplementary Table 4.

Hit-processing approximation

During hit processing, 2 algorithms may be used, the DFS, and, as a backup (if the DFS algorithm's runtime exceeds 60 seconds), the heuristic. We calculated how many times the heuristic algorithm was used as a backup during the hit processing of the 2010–2020 UniProt sample. We found that for the intra-HMM hit processing, the heuristic algorithm was used in 7.2% of the sequences, and for the inter-HMM hit processing in 0.5% of the sequences.

Quality control with sequenced organisms

As a secondary quality control, to assess the impact on F1 score when using taxon-resolved reference data, we annotated several sequenced organisms (for more details, see Supplementary Table 5) with and without TSHMMs. We also evaluated the impact of the different hit-processing algorithms on these samples. As seen in Fig. 4, well-studied organisms (e.g., *S. cerevisiae*) had better annotations, especially when applying TSHMMs, unlike poorly described organisms. The average F1 score gain with TSHMMs was 0.006. With TSHMMs, the DFS algorithm had, on average, 0.001 and 0.010 higher F1 scores than the heuristic and BPO algorithms, respectively. Without TSHMMs, the DFS algorithm had, on average, 0.008 and 0.013 higher F1 scores than the heuristic and BPO algorithms, respectively. Further details can be found in Supplementary Table 6.

Comparison between Mantis and other PFA tools

The sequenced organisms enumerated in Supplementary Table 5 were annotated with Mantis, eggNOG-mapper, and Prokka (for the latter non-prokaryote organisms were excluded). To evaluate the added value of using the very comprehensive eggNOG reference data source, we also assessed Mantis's F1 score using different reference data. In total, 6 different tests were performed for each organism: (i) Mantis with default data sources and with taxonomy information, (ii) Mantis with default data sources except for eggNOG's data and with taxonomy information, (iii) Mantis with default data sources but without taxonomy information, (iv) eggNOG-mapper without tax scope option, (v) eggNOG-mapper with tax scope option, and (vi) Prokka with default data sources and default execution.

On average, test (i) had F1 score and annotation coverage of 0.857% and 96.56%, respectively; (ii) 0.832% and 89.82%; (iii) 0.850% and 96.14%; (iv) 0.734% and 88.45%; (v) 0.725% and 88.02%; and (vi) 0.507% and 62.38%. As seen in Fig. 5, Mantis outperformed the other PFA tools in all tests (with 1 exception in the organism *S. cerevisiae*, where eggNOG-mapper without taxonomy had an F1 score of 0.841 and Mantis without taxonomy had an F1 score of 0.830). The mean Mantis F1 score with default execution and TSHMMs was 0.131 higher than eggNOG-mapper (with tax scope) and 0.360 higher than Prokka. Mantis's setting without the eggNOG reference data had a mean F1 score 0.107 higher than eggNOG-mapper (both tools with taxonomy information) and a mean F1 score 0.025 lower than Mantis's with the eggNOG reference data. Further details are available in Supplementary Table 7.

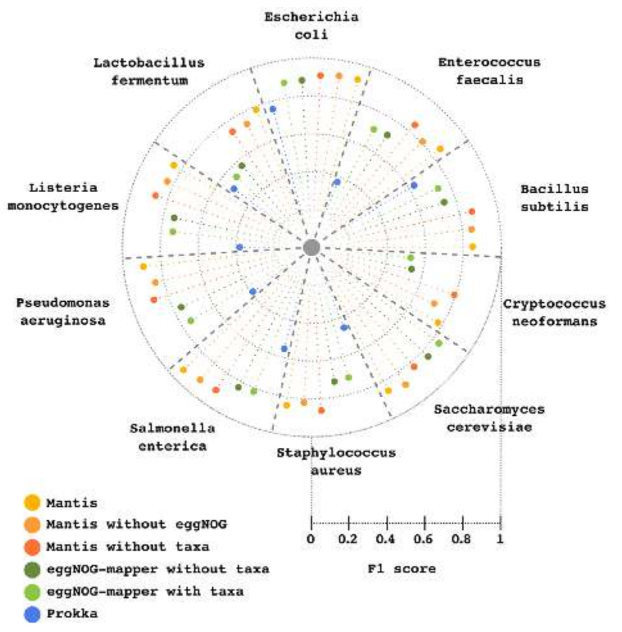

Figure 5: Annotation F1 score of Mantis, eggNOG-mapper, and Prokka using different reference data. Each slice represents an organism and contains the F1 score obtained between the different conditions.

Annotating metagenomes

To our knowledge, there are no manually curated metagenome annotations, therefore annotation validation was not performed; instead we only calculated the annotation coverage. We selected 4 samples from different environments and predicted the protein-coding genes with Prodigal v2.6.3 [60]. The annotated samples were:

- Biogas highly efficient cellulose-degrading consortium (SEM1b) [61, 62] with 39,411 sequences;
- Glacier-fed stream sediment (GFS) [63] with 270,341 sequences (phenol-chloroform extraction batch No. 37);
- Marine [64] with 605,043 sequences (ERR1726751);
- Human gut microbiome (MuSt [7]) with 692,061 sequences (M05-01-V1).

The performance of Mantis varied per metagenome sample; it annotated 213,539, 162,133, 33,016, and 559,792 sequences in the samples GFS, marine, SEM1b, and MuSt, respectively. The respective annotation coverage was as follows: 78.99%, 26.80%, 83.77%, and 80.89%. We repeated the same test for eggNOG-mapper and Prokka (in the case of Prokka by annotating the original nucleotide sequences); the coverage for the samples GFS, marine, SEM1b, and MuSt, was, respectively, 77.52% and 10.87%, 16.21% and 1.01%, 81.95% and 32.32%, and 78.72% and 20.37%.

Computational efficiency

We ran Mantis against samples with a different number of sequences and a different number of available CPUs. We performed this test for the DFS and heuristic algorithm only. As expected, we found that the heuristic algorithm was faster than the DFS algorithm. The heuristic algorithm was, on average, 1.42 times faster than the DFS algorithm. As expected, runtimes were inversely correlated to the number of CPUs and sequences. Further details can be found in Supplementary Table 8.

We also aimed at allowing Mantis to be run on personal computers, which requires removing the eggNOG dataset. However, as we have previously shown in "Comparison between Mantis and other PFA tools," this does not have a large effect on F1 score. We annotated the previously enumerated sequenced organisms (Supplementary Table 5) on a Dell XPS 13-9370 with Ubuntu 20.04.1 LTS 64 bit, 16 GB RAM, 512 GB SSD, and an 8 core Intel Core it-8550U CPU. The mean runtime for prokaryotes and eukaryotes was 28 and 93 minutes, respectively. Further details are available in Supplementary Table 9.

Discussion

We herein presented Mantis, an open-access PFA tool that produces high-quality annotations and is easily installed and integrated into other bioinformatic workflows. Mantis uses a well-established homology-based method and produces high-quality consensus-driven annotations by relying on the synergy between multiple reference data sources and improved hit-processing algorithms.

Mantis addresses some major challenges in PFA, such as flexibility, speed, the integration of multiple reference data sources, and use of domain-specific annotations. It also addresses under-annotation through the use of multiple reference data sources, which implicitly leads to a wider search space. Additionally, redundancy, which is a drawback inherent to consensus-driven annotation, is ameliorated by removing duplicate database IDs and/or identical descriptions. We have attempted to avoid over-annotation through the generation of a consensus-driven annotation, which identifies and merges annotations that are consistent (i.e., similar function) with each other (e.g., if 3 of 5 independent sources point towards the same function and 2 others point towards other, unrelated functions, then these 3 annotations are more likely to be valid), and eliminating the remaining inconsistent annotations.

We have shown that a stricter/lower e-value threshold did not necessarily lead to a higher F1 score. As expected, a lower threshold restricted the amount of hits, lowering the recall. However, we also found that more stringent e-value thresholds may result in a lower precision; this behaviour is connected to Mantis's consensus generation and hit combination scoring. A thorough explanation is available in the Supplementary PDF.

Well-curated and commonly used resources were chosen as the default reference data sources for Mantis, containing both unspecific and specific reference data (e.g., taxon-specific). As we have shown, no single reference data source accounted for most annotations, each offering both unique and overlapping insight into protein function, thus confirming their synergy and partial redundancy. These are integrated through a consensus-driven approach, which Mantis uses as an additional quality control step, and a means to automatically incorporate a broader variety of IDs. The intersection of IDs was, as expected, the main contributor towards this integration (because most databases provide cross-linking); however, we found that the text-mining approach still contributed considerably (35.12% for the UniProt 2010-2020 sample), which clearly highlights the need to use such a method.

We additionally evaluated the impact of not using text mining during consensus generation and removing the consensus generation altogether on the DFS and BPO algorithms. The benchmark using the BPO algorithm without consensus generation represented the baseline approach towards the integration of multiple reference data sources (merely selecting the most

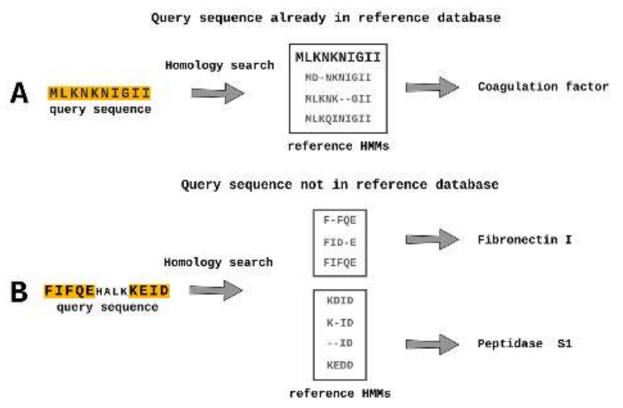

Figure 6: The impact of the reference data completeness on protein function annotation. **A.** The functional prediction is facilitated by the query sequence being previously identified and included in the reference HMMs. **B.** If the query sequence has not been previously annotated, multiple regions in the protein may match with different reference HMMs.

significant hit during inter- and intra-HMM hit processing). In contrast, the benchmark using the DFS algorithm with the consensus generation depicted the accumulation of all the features introduced by Mantis. Overall, we found a difference of 0.064 in F1 scores, which suggests the additive effect of Mantis's various data integration methods. Mantis, in respect to this specific benchmark, also obtained an F1 score higher than eggNOG-mapper in all conditions, which suggests the importance of using multiple reference data sources.

We have implemented 2 algorithms for domain-specific homolog search (DFS and heuristic as backup) and have not only shown that these algorithms perform better when annotating previously described protein sequences but that their impact on the F1 score increased when annotating previously uncharacterized protein sequences (e.g., average F1 score gain with DFS and BPO algorithms in the UniProt 2010–2020 and 2020 samples was 0.013 and 0.033, respectively). We hypothesize that for the latter, a homology search is not capable of finding whole-sequence homologs, finding, however, multiple domains that partially constitute the protein sequence. As such, we argue that by increasing the resolution (sequence homology to domain homology) of homology-based reference data, domain-specific algorithms may become increasingly valuable. We think that this would be especially important when annotating protein sequences without well-described homologs but that contain previously characterized conserved protein domains. In Fig. 6A, we can observe that the present query sequence is already used to generate the HMM profiles in the reference data, matching with the HMM profile containing it. Such a scenario is common when annotating well-described organisms (e.g., *Escherichia coli*). However, as is often the case when annotating non-model organisms and metagenomes, the query sequence is absent from the reference data (Fig. 6B), thus partially matching with several HMMs (which may correspond to multiple domains, depending on the resolution of the reference data). Unlike the BPO algorithm, the heuristic and DFS algorithms are able to incorporate multiple homologs. While these may not be enough to determine a protein's biological function, they still provide a better biological context than a single functional annotation.

Further improvements in annotation quality may also require the use of motif-based and/or genomic context–based (e.g., operon context information, co-expression, and subsystems) methods such as those described by Sigrist et al. [65], Mooney et al. [66], Mavromatis et al. [67], Overbeek et al. [21], and Hannigan et al. [68]. Nevertheless, the significantly higher F1 score seen when comparing the DFS and BPO algorithms highlights the need to adopt better hit-processing methods, especially for non-model organisms. With samples ranging from thousands to millions of protein sequences, sub-optimal hit-processing algorithms may cascade into unnoticeable pitfalls in downstream data analysis (e.g., accumulation of incomplete or low-quality genome annotation, which may lead to false biological interpretations). While we have shown that the DFS algorithm outperforms the heuristic algorithm, both achieve a very similar F1 score when applied to non-synthetic samples; because the heuristic algorithm is much more time efficient (as seen in Supplementary Table 8), a user may confidently set it as primary algorithm.

The use of TSHMMs resulted in a 0.006 higher F1 score; however, this improvement (as seen in Fig. 4) was not consistent across all the annotated organisms (as expected, a similar trend was also seen with eggNOG-mapper). We believe that this is due to a poorer quality of the TSHMMs for some organisms, which is a consequence of the issues with the current taxonomy classification system [69, 70] and lack of knowledge regarding highly resolved taxa [71]. Model organisms such as *E. coli* and *S. cerevisiae* clearly benefited from TSHMMs, both because the reference data already contain data specific to these organisms and because functions of proteins within model organisms are better experimentally described. Conversely, non-model organisms are often only computationally annotated by association, contributing to a weaker reference annotation (which can be observed by the higher rate of potentially new annotations in these organisms, as seen in Supplementary Table 6). Nonetheless, while experimental evidence remains the gold standard, it is unfeasible to ignore the need for computational methods to infer function. While steps in this direction have been taken [16, 57], taxon-resolved PFA remains a challenge.

We benchmarked Mantis against 2 other PFA tools—eggNOG-mapper and Prokka—and have shown that Mantis achieves a higher F1 score (0.131 higher than eggNOG-mapper and 0.350 higher than Prokka). Although Mantis's default execution heavily relies on the eggNOG reference data, we have also shown that even without it, it is possible to achieve an almost similar F1 score. This attests to the quality of the various reference data used, showcasing as well the possibility of running Mantis on a personal computer (something that would be impossible with eggNOG's prohibitive size).

We also evaluated the annotation coverage of Mantis and the other PFA tools when annotating metagenomes. Mantis had the highest annotation coverage among the tested PFA tools, but eggNOG-mapper was close behind. All PFA tools had a low annotation coverage for the marine sample. We believe that this may be due to a lack of reference HMMs for this specific environment. This metagenomic sample has data from varying ocean depths, with many novel sequences from viruses, prokaryotes, and picoeukaryotes [64].

Finally, as shown in "Accessibility and scaling," a conda environment and automated reference data download are provided. In addition, Mantis accepts several formats as input (i.e., protein FASTA file, TSV file with paths, directories, or compressed archives), outputting easy-to-parse TSV files. We believe that these features address some of the reproducibility challenges that the bioinformatics community still faces [72].

As discussed, there is still room for improvement in the hit-processing algorithm DFS (because it does not provide large F1 score gains over the heuristic algorithm). In the future, Mantis could also include genomic context–based annotation methods. Despite the aforementioned challenges, we have clearly shown that Mantis is a flexible tool that also produces annotations with high precision and recall.

Conclusion

By making use of the synergistic nature of differently sourced high-quality reference data, Mantis produces reliable homology-based annotations. By allowing for total customization of these reference data, Mantis is also flexible, easily integrated and adapted towards various research goals. In conclusion, we have shown that Mantis addresses a number of the current PFA challenges, resulting in a highly competitive PFA tool.

Methods

Accessibility and scaling

Mantis automatically sets up its reference data by downloading HMMs from different sources and, when necessary, reformatting the data to a standardized format and downloading any relevant metadata. Reference data can be customized via a config file. It also dynamically configures its execution depending on the resources available. A conda environment and extensive documentation [54] are available.

Mantis splits most of the workflow into sub-tasks and subsequently parallelizes them by continuously releasing tasks to workers from a global queue (via Python's multiprocessing module). During each main task of the annotation workflow, workers are recruited (the number of workers depends on the available hardware and work required); these will then execute all the queue tasks. When a worker has finished its job, it will execute another task from the queue until there are no more tasks to execute. If the queue is well balanced, minimal idle time (time spent waiting for workers to get a new task) can be achieved. Load balancing is achieved by splitting the sample and reference data into chunks. During set-up, large reference data sources (>5,000 HMM profiles) are split into smaller chunks; this enables parallelization and ensures that each annotation sub-task takes approximately the same time. Samples are equally split into chunks (sample chunk size is dynamically calculated). If the sample has ≤200,000 sequences, sequences are distributed by their length among the different chunks, so that each chunk has approximately the same number of residues. If the sample has >200,000 sequences, then sequences are distributed to each chunk independently of their length (this alternative method is an efficiency safeguard). This 2-fold splitting achieves quasi-optimal load balancing. With the sample and reference data in chunks, posterior workflow steps can be parallelized wherever applicable. It is noteworthy that Mantis uses HMMER's hmmsearch for homology search, which outputs an e-value scaled to the sample/chunk size. Because Mantis splits the samples into chunks, during hit processing, the e-value is scaled to the original sample size.

Input and output

MANTIS accepts protein sequence FASTA files as input. If the sample has been previously taxonomically classified, the user can add this information when running Mantis. For example, if annotating an *E. coli* sample, the user could add "–od" followed by the NCBI ID or the organism name:

```
$ python mantis run_mantis -t sample.faa -od 562
```

Mantis outputs, for each sample, 3 TSV files, each corresponding to a different step in Mantis's workflow: (i) a raw output output_annotation.tsv (generated during Fig. 1 step "Intra-HMM hits processing"), with all the hits, their e-value, and coordinates; (ii) integrated_annotation.tsv (generated during Fig. 1 step "Metadata integration"), with the same information as output_annotation.tsv, but also with hits metadata (e.g., KEGG orthology IDs [KO], enzyme commission [EC] numbers, free-text functional description); and (iii) the main output file consensus_annotation.tsv (generated during Fig. 1 step "Consensus generation"), with each query protein ID and their respective consensus annotation from the different reference data sources (e.g., Pfam). These files provide contextualized output in a format that is both human and machine-readable. A Mantis.out file is also provided per sample, serving as a log file for each execution step.

Reference data and customization

Mantis, by default, uses multiple high-quality reference HMM sources: Pfam [56], eggNOG [57], NPFM [58], KOfam [55], and TIGRfam [59] (these default HMMs can be partially or entirely removed). To find more meaningful homologs through TSA, Mantis uses TSHMMs, originally compiled by eggNOG and NPFM. The eggNOG TSHMMs were compiled by downloading all the TSHMMs at http://eggnog5.embl.de/download/latest/per_tax_level/; their respective metadata originate from the metadata available in the aforementioned link, as well as the metadata within the eggNOG-mapper SQL database. NPFM TSHMMs were compiled by downloading all the NPFM HMMs at https://ftp.ncbi.nlm.nih.gov/hmm/current/ and assigning each HMM into their respective TSHMM. A general NPFM HMM was created by pooling all non-assigned HMM profiles and the TSHMMS from the following NCBI IDs: 2157 (*Archaea*), 2 (*Bacteria*), 2759 (*Eukaryota*), 10239 (*Viruses*), 28384 (Others), and 12908 (Unclassified). These IDs correspond to NCBI's top-level taxonomy rank IDs. A general eggNOG HMM was created by pooling together the TSHMMs from the same aforementioned NCBI taxon IDs. The user can customize which eggNOG TSHMMs are downloaded by Mantis by adding the line "nog_tax = NCBI_ID1, NCBI_ID2" to the config file. Custom HMM sources can also be added by the user; metadata integration of these is also possible (an example is available in Mantis's repository). Because some sources are more specific than others, the user can also customize the weight given to each source during consensus generation. HMM profiles often only possess an ID respective to the database from which they were downloaded, which may not directly provide any discernible information. Mantis, when necessary, ensures that the hits from these HMMs are linked to their respective metadata. For future reference, while an HMM is an individual profile, Mantis compiles all related HMM profiles into a single file, making it indexable by HMMER. Thus when a certain HMM source is mentioned, it refers to the collection of related HMM profiles.

Taxon-specific annotation

TSA uses the TSHMMs and unspecific HMM made available by eggNOG and NPFM. TSA, however, works differently from the annotation method of the other reference data. When given taxonomy information (either a taxon name or NCBI ID) the organism's taxonomic lineage is computed (e.g., for *E. coli* the lineage would be "2 - 1224 - 1236 - 91347 - 543 - 561 - 562"). TSA starts by searching for homologs in the most resolved TSHMM (in this case for taxon 562, if it exists). All valid homologs (respecting the e-value threshold) are extracted for each query sequence, and unannotated sequences are compiled into an intermediate FASTA file. A new homology search round starts with the sequences in the current intermediate FASTA, but now in the TSHMM 1 level above (in this case the TSHMM 561). This cycle repeats until all query sequences have valid homologs or until there are no more TSHMMs to search for. If there are still sequences to annotate, then these homologs are searched for in the general eggNOG and NPFM HMMs. If no taxonomy information is given, the homology search starts with the general NPFM and eggNOG HMMs. Non-taxon-specific HMMs (i.e., Pfam, KOfam, and TIGRfams) are always used, regardless of the sample's taxonomy.

Multiple hits per protein

HMMER outputs a "domtblout" file [24], where each line corresponds to a hit/match between the reference data and the query protein sequence. The e-value threshold within the HMMER command limits the amount of hits to be analysed in the posterior processing steps. Each hit, among other information, contains the coordinates where the query sequences matched with the reference HMM profiles and the respective confidence score (e-value) (Fig. 2A and B). Mantis uses HMMER's independent e-value when using the DFS and heuristic algorithms, whereas it uses the full sequence e-value when using the BPO algorithm (because only the best hit is extracted per protein sequence). For simplicity purposes, both are simply referred to as e-value throughout this article. The annotation of a protein sequence with multiple hits is a nontrivial problem, thus requiring the implementation of a method for the processing of hits. We designed a method that generates and evaluates all possible combinations of hits by applying the DFS algorithm [73]. This algorithm allows the traversal of a tree-structured search space (i.e., each node is a hit), whilst pruning solutions that do not respect predefined constraints (i.e., overlapping hit residue coordinates), backtracking from leaf to root until the possible solution space is exhausted. Our method generates all the possible combination hits with the following method: (i) get 1 hit from the collection of hits and define it as the combination root hit; (ii) check which other hits overlap up to 10% (default value) [31] with previous hits and select 1 to add to our present combination of hits; (iii) repeat step (ii) until no more hits can be added; (iv) repeat steps (i–iii) so that we loop over all the other hits and all possible combinations are generated. We used Cython [74] to speed up the DFS implementation. Cython is an optimizing static compiler for the Python programming language, allowing the compiler to generate C code from Cython code, in this case, functioning as a wrapper for the DFS algorithm. The total number of possible combinations is $2^N - X - 1$, where N is the number of hits the protein sequence has, X the number of impossible combinations (combinations with overlapping hits), and 1 the empty combination. Owing to exponential scaling, this method is not always computationally feasible (e.g., the query sequence is very large and has many small-sized hits). In such a scenario, the DFS algorithm may exceed the system's recursion limit or be unable to find a solution in optimal time (60 seconds by default, but customizable). Should this happen, Mantis uses the previously described heuristic algorithm, which scales linearly (a warning is written in the Mantis.out log).

After generating all the possible combinations, each combination is evaluated according to several parameters:

- $\text{query}_{\text{length}}$—number of residues in the query sequence.
- $\text{hit}_{\text{length}}$—number of residues in the hit.
- $\text{combo}_{\text{length}}$—number of hits in the respective combination.
- Total coverage (TC)—number of non-redundant residues in all the combination's hits divided by $\text{query}_{\text{length}}$. A high TC implies that the combination covers a large percentage of the protein sequence.
- Average hit coverage (HC)—sum of the coverage of each hit ($\text{hit}_{\text{length}}/\text{query}_{\text{length}}$). This sum is then averaged by dividing by $\text{combo}_{\text{length}}$. A high HC implies that the hits in the combination are large, thus benefiting combinations with few large hits rather than combinations with many small hits.
- Combination e-value (CE)—the e-value of each hit is scaled twice, once to reduce the range between different e-values ($\log_{10}$) and the second time to understand how each hit e-value compares to the best/lowest hit e-value found for a particular sequence (minmax scaling). The scaled e-values are then summed and divided by $\text{combo}_{\text{length}}$.

The "combination score" is defined by the following equation:

$$\text{TC} \times \text{HC} \times \text{CE}. \tag{1}$$

The combination with the highest combination score is then selected, where the available choices will ultimately depend on the algorithm used (Fig. 2c). Our intra-HMM hit-processing implementation thus applies a 2-fold quality control, initially by limiting the amount of hits in HMMER's domtblout (i.e., e-value threshold) and second by hierarchically ordering and selecting the most significant combination of hits.

Using multiple reference data sources

An unannotated protein sequence may match with 0, 1, or multiple reference HMM profiles, from 1 or more data sources. When a protein sequence has multiple hits from different data sources, it is important to identify functionally similar annotations so that no information is lost (i.e., functional descriptions or IDs that may be in 1 reference data source but not in another). By linking the metadata respective to the HMM profiles to the now annotated protein sequence, we can identify functionally similar annotations and integrate multiple reference data sources into 1 final consensus annotation. In this manner, functionally similar annotations are merged, and any complementary information they provide can then be used in downstream analysis (e.g., Annotation 1 has a Pfam and KO ID, Annotation 2 has an EC number and the same KO ID; merging these will result in a final annotation with more information).

For the integration of functional annotations from multiple data sources, a two -fold approach was used: (i) consensus between IDs and (ii) consensus between the free-text functional description. The latter is used as a backup because ID cross-linking is not universally available. Each reference data source includes metadata relevant to the HMM profiles herein; these metadata may include multiple intra- and/or inter-database IDs, as well as free-text functional descriptions. IDs are extracted either through source-specific metadata parsing or regular expressions. Free-text functional descriptions are extracted by source-specific metadata parsing. With this information it is then possible to identify annotations that are functionally similar/consistent and may thus be complementary to each other. The consensus between IDs is calculated by identifying intersections between the functional annotations of different reference data sources (e.g., both annotations have the same Pfam ID). IDs

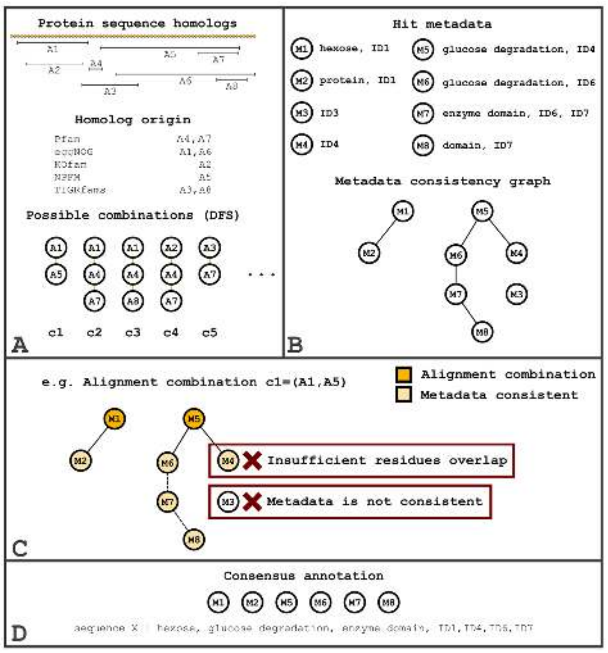
Protein sequence homologs
A1
A2
A3
A4
A5
A6
A7
A8
Homolog origin
Pfam A4, A7
eggNOG A1, A6
KOfam A2
NPFM A5
TIGRfams A3, A8
Possible combinations (DFS)
c1 c2 c3 c4 c5
A
Hit metadata
M1 hexose, ID1
M2 protein, ID1
M3 ID3
M4 ID4
M5 glucose degradation, ID4
M6 glucose degradation, ID6
M7 enzyme domain, ID6, ID7
M8 domain, ID7
Metadata consistency graph
B
e.g. Alignment combination c1=(A1,A5)
Alignment combination
Metadata consistent
Insufficient residues overlap
Metadata is not consistent
C
Consensus annotation
sequence X | hexose, glucose degradation, enzyme domain, ID1,ID4,ID5,ID7
D

Figure 7: Inter-HMM hit-processing steps. Inter-HMM hit processing starts by pooling all hits [A1, AN] together (regardless of the reference data source) and generating all the possible (non-overlapping coordinates) combinations [c1, cN] (A). A metadata consistency graph (B) is also built by connecting all nodes [M1, MN] that have intersecting IDs or highly similar descriptions (e.g., A1's metadata M1 is consistent with A2's metadata M2 (shared ID1), and A5's metadata M5 is consistent with A6's metadata M6 (similar description "glucose degradation"). With this metadata consistency graph, the hit consistency HCN score of each combination is calculated. For c1, for example, a sub-graph containing M1, M5, and all directly connected nodes (only M2 and M6 but not M4 because it has insufficient residue overlap—A4) would be created. The number of nodes in this sub-graph would then be divided by the total number of nodes in the original graph; therefore c1 would have an HCN of (2 + 2)/8 = 0.5. The remaining parameters would then be calculated and the best combination, according to equation 2, would be selected. Finally, if, for example, the best combination is c1, then this combination is expanded by merging all nodes directly or indirectly connected to M1 and M5 in the metadata consistency graph (C) and with sufficient residue overlap (i.e., M2, M6, M7, M8). The expanded combination is then merged into the final consensus annotation (D).

within the free-text functional descriptions are extracted (with regular expressions) and also used here. If no consensus between IDs is found, then we proceed with a consensus calculation between functional descriptions (further described in the Supplementary PDF).

Inter-HMM hit processing starts by pooling together all hits from the different reference data sources and generating all possible combinations of hits (Fig. 7A). The same method used in intra-HMM hit processing is applied, where the DFS algorithm is used by default (again using the heuristic algorithm as a backup), but the BPO and heuristic algorithms can also be used. We then check the metadata consistency (either through IDs or free-text functional descriptions) of each hit against the current sequence's other hits. With this information, a metadata consistency graph is generated (Fig. 7B). With the metadata consistency graph and all possible combinations of hits, we can then calculate the consensus combination score using equation 2. This requires calculation of the combination score, using equation 1. This score is then multiplied by an additional score, comprising the following parameters:

- Average hit consistency (HCN)—number of hits (among all hits) with metadata directly consistent (i.e., nodes directly connected in the metadata consistency graph) to the hits in the present combination. Consistency checks are restricted to other reference data sources besides the hit own's reference source (e.g., if a hit is from Pfam, we would only check hits that are not from Pfam). This number, plus the number of hits in the combination, is divided by the total number of hits for the respective query sequence [e.g., if a combination has 2 hits, with these having metadata consistent with 3 other hits, and if there are 10 hits in total, HCN would be equal to (2 + 3)/10 = 0.5]. This is an important parameter because it entails independent sources describing the same function.
- Reference HMM weight (HMMW)—mean weight of all the reference data sources within the combination. This is calculated by adding all hits' HMM weights and dividing this sum by the number of hits in the combination [e.g., if a hit comes from Pfam that has a weight of 1, and another from eggNOG that has a weight of 0.8, HMMW would be equal to (1 + 0.8)/2 = 0.9]). The default weight for each default reference data source has been set according to the authors' perception of the reference quality—creation method, curation level, and annotation completeness (eggNOG, 0.8; Pfam, 0.9; NPFM and KOfam, 0.7; and TIGRfam, 0.5). This weight is customizable; the default weight for custom reference data is 0.7 (which can also be customized).
- Metadata quality (MQ)—mean metadata quality of each hit in the combination. If a hit has no annotation data (IDs or description), it is given a score of 0.25; 0.5 if only the description; 0.75 if only the IDs; 1 if IDs and description. All hits' metadata quality score is summed and divided by the number of hits in the combination.

Note that hit metadata consistency (through IDs or descriptions) requires a minimum of 70% residue overlap (default but can be changed). Using the previously calculated combination score, we then calculate the consensus combination score using the following equation:

$$\text{Combination}_{\text{score}} \times \frac{\text{HCN} + \text{HMMW} + \text{MQ}}{3}. \qquad (2)$$

The combination with the highest consensus combination score is selected and expanded by concatenating additional metadata from other consistent hits (Fig. 7C). In this step, consistent hits can be either directly or indirectly connected in the metadata consistency graph (a minimum of 70% residue overlap is still required). This expanded combination is then merged into the final query sequence consensus annotation (Fig. 7D). Redundant (i.e., repeated identifiers or functional descriptions) or poor-quality information (e.g., "hypothetical protein") is removed from the consensus annotation.

Sample selection

To select an initial testing dataset we started by downloading all the curated Uni-Prot [75] (i.e., Swiss-Prot) protein entries created after 2010 (until 14 April 2020), along with their respective sequences, annotations, and annotation scores. We then split these entries by date, 2010–2020, 2015–2020, 2018–2020, and 2020 only. For genomic sample benchmarking we selected organisms widely used in microbial community standards. The respective genomes, proteomes, and reference annotations were then downloaded from Uniprot on 26 May 2020 (Supplementary

Table 5). These samples were also used for comparing Mantis to eggNOG-mapper and Prokka.

Establishing a test environment

For annotation quality benchmarking, we evaluate each annotation produced by Mantis and check whether it agrees (database IDs intersection) with the respective reference annotation, creating a confusion matrix. We created 2 main types of test samples, the first consisting exclusively of curated UniProt [75] protein entries (and the respective annotations), which were then split by date of creation (2010–2020, 2015–2020, 2018–2020, 2020); and the second type consisting of organism-specific UniProt protein entries, with a mix of curated and automatically generated annotations. Each sequence's reference annotation consists of the UniProt protein function annotations. Each sequence reference annotation and the respective PFA tool's annotation is composed of a set of identifiers (if available: enzyme ECs, Gene Ontology (GO) IDs, eggNOG IDs, KEGG orthology IDs, Pfam IDs, and TIGRfam IDs) and functional descriptions. During the benchmark process, each sequence's reference annotation (e.g., "glucose degradation ID1") is compared against the PFA tool (i.e., Mantis, eggNOG-mapper, and Prokka) annotation (e.g., "degrades glucose ID1"). This comparison entails checking whether any of the database IDs present in the reference annotation (i.e., ID1) are also present in the PFA tool annotation (i.e., ID1); if they are, we consider this annotation to be the same. This has some significant limitations: (i) the functional description is the same but the corresponding set of identifiers is not; and (ii) when annotating multiple regions of the protein (which is the case when using Mantis's DFS and heuristic algorithms), it is possible that only 1 of the annotated regions has IDs that intersect with the respective sequence reference annotation. Unfortunately, owing to the different resolutions of the reference HMMs, it is not always possible to understand whether an annotation refers to a specific domain or a partial whole-sequence hit. While a domain-centric benchmark would be feasible for Pfam, the same is not true for the remaining reference HMMs with broader resolutions (e.g., TIGRFams provides general functional annotations). However, as we have previously shown, even when using the BPO algorithm, Mantis has shown to output almost equally high F1 scores. Despite these limitations, because whole-sequence reference annotations contain comprehensive cross-linking with other databases, it provides clear benefits: (i) it fits better for the wide-ranging scopes of the reference data sources, and (ii) it allows for a more fair benchmark of the different PFA tools that may use different reference data sources (and thus output annotations with different database IDs). This method then allows for the construction of a confusion matrix, where each pairwise whole-sequence annotation comparison (PFA tool/reference annotation) corresponds to a single class. TPs occur when the PFA tool–generated annotation and the reference annotation share 1 or more database IDs (e.g., Pfam ID), and FPs when no database IDs are shared. FNs occur when the PFA tool does not annotate a protein sequence, although a reference annotation is available; and TNs when the PFA tool does not annotate a protein sequence, and no reference annotation is available. The functional text descriptions are not taken into account during the benchmark; therefore if an annotation has no IDs, we simply consider there to be no annotation. Protein sequences annotated with the descriptions "unknown function," "uncharacterized protein," "hypothetical protein," or with Pfam's "domain-unknown-function"/DUF IDs are not taken into account during benchmarking (for reference and PFA tool annotations). In addition, it is also possible for the reference or PFA tool not to have an annotation for a certain sequence. In any of the these 3 scenarios, if the PFA tool manages to annotate the sequence, this case is classified as a potentially new annotation (PNA). Because no ground-truth exists in these scenarios, PNAs are excluded from the confusion matrix classes (not used during any performance metrics) and are only used to calculate the annotation coverage. PNAs can potentially provide novel insight into protein sequences without any previous annotation. Because, by default, most sequences used during benchmarking will have an annotation, TNs, and ergo any metrics using TNs (e.g., specificity), are irrelevant.

"Annotation coverage" is defined here as the number of annotations produced by the PFA tool divided by the total number of protein sequences in a sample $\text{Total}_{\text{seqs}}$. $\text{Total}_{\text{seqs}}$ includes sequences with and without a reference annotation (because not all sequences have a reference annotation); the total number of the PFA tool annotations includes TPs, FPs, and PNAs. Annotation coverage is calculated by $(\text{TP} + \text{FP} + \text{PNA})/\text{Total}_{\text{seqs}}$. Numerous metrics can be calculated from the various confusion matrix categories; we considered precision and recall/sensitivity to be among the most important. Precision is defined as $\text{TP}/(\text{TP} + \text{FP})$ and corresponds to the number of correctly annotated protein sequences out of all the protein sequences that the PFA tool managed to annotate. Recall is defined as $\text{TP}/(\text{TP} + \text{FN})$ and corresponds to the number of correctly annotated protein sequences out of all the protein sequences that we know the function of (i.e., protein sequences that have a reference annotation). Both are equally important; a tool with low precision will incorrectly annotate protein sequences, whereas a tool with low recall will not produce sufficient annotations. A way to converge both scores into 1 is to use the F1 score, which is defined as $2 \times [(\text{Precision} \times \text{Recall})/(\text{Precision} + \text{Recall})]$. Unless otherwise stated, values shown in this article are shown as absolute values ranging from 0 to 1.

Finally, we benchmarked Mantis against 2 other PFA tools, eggNOG-mapper and Prokka. For homology search, Mantis uses HMMER [24], for eggNOG-mapper we used the Diamond-based [23] search (as suggested by the authors), and Prokka uses BLAST and HMMER.

All tests ran on high-performance computing resources with Dell C6320, 2 ∗ Intel Xeon E5-2680 v4 at 2.4 GHz [76]; each core had 4 GB of RAM. Unless specified, all tests ran with 25 cores and 100 GB RAM (actual Mantis minimum hardware requirements are much lower). In addition, the same methodology and nomenclature apply to any other benchmarked tools described in this article. Mantis used HMMER v3.2.1. The local version of eggNOG-mapper used was v2.0.6 with database v5.0.1 found at https://github.com/eggnogdb/eggnog-mapper/commit/41ec3566ab00fd437f905dfde592c553632a9eae. The local version of Prokka used was v1.14.6 found at https://github.com/tseemann/prokka/releases/tag/v1.14.6. For details on execution commands see the Supplementary PDF.

Testing different e-value thresholds

Different e-value thresholds were tested: $1e^{-3}$, $1e^{-6}$, $1e^{-9}$, $1e^{-12}$, $1e^{-15}$, $1e^{-18}$, $1e^{-21}$, $1e^{-24}$, $1e^{-27}$, $1e^{-30}$, and a dynamic threshold. The dynamic threshold was set according to the query sequence length, which was previously shown to provide better results with BLAST [34]. For the dynamic threshold, for sequences with <150 amino acids, the e-value threshold was set to $1e^{-10}$; if >150 and <250, $1e^{-\text{sequence}_{\text{length}}/10}$; and if >250, $1e^{-25}$. The UniProt 2010–2020 sample was then annotated with all the different e-value thresholds, and each output was compared to the reference annotations.

Testing hit-processing algorithms

To understand whether the different hit-processing algorithms resulted in statistically significant differences in F1 scores, we created 5,000 randomized synthetic samples with 5,000 sequences each, which were randomly selected from the 2010–2020 UniProt sample. Per algorithm, we compared the Mantis annotations of each subset to the reference annotations (to allow for pairwise comparison of each algorithm, the same subsets were used in all algorithms). This resulted in a list of confusion matrices (5,000 per algorithm), from which we calculated the F1 score. We applied the Wilcoxon signed-rank test, with the H_0: no differences in F1 score between the tested algorithms. As a non-parametric test, this test makes no assumptions on the distribution of the data. A pairwise comparison was done between DFS and the other algorithms: (i) DFS and heuristic and (ii) DFS and BPO.

Availability of Source Code and Requirements

- Project name: Mantis
- Project home page: https://github.com/PedroMTQ/mantis
- Operating system: Linux
- Programming language: Python
- Other requirements: Python 3+, HMMER 3+, and several Python packages (see the provided environment for a full list)
- License: MIT
- RRID:SCR_021001
- Biotools ID: mantis_pfa

Data Availability

The data and code supporting the results of this article are available at https://git-r3lab.uni.lu/pedro.queiros/mantis_supplements. An archival copy of the code and supporting data is available via the *GigaScience* repository, GigaDB [77].

Additional Files

Supplementary pdf. (i) discussion on how the e-value threshold may change Mantis' output, (ii) execution commands, and (iii) information on how the similarity analysis was performed.
Supplementary Table 1. Function assignment e-value threshold
Supplementary Table 2. Impact of hit processing algorithms
Supplementary Table 3. Impact of sample selection
Supplementary Table 4. Impact of consensus generation
Supplementary Table 5. Quality control against sequenced organisms – list of samples
Supplementary Table 6. Quality control against sequenced organisms – results
Supplementary Table 7. Comparison between Mantis and other PFA tools
Supplementary Table 8. Annotation efficiency – random sequences
Supplementary Table 9. Annotation efficiency – personal PC
Supplementary Table 10. Metagenome coverage

Abbreviations

BLAST: Basic Local Alignment Search Tool; BPO: best prediction only; CE: combination e-value; CPU: central processing unit; DFS: depth first search; EC: enzyme commission; FP: false positive; FN: false negative; GFS: glacier-fed stream sediment; GO: gene ontology; HC: average hit coverage; HCN: hit consistency; HMM: hidden Markov models; HMMW: average reference HMM weight; ID: database identifier; KEGG: Kyoto Encyclopedia of Genes and Genomes; KO: KEGG orthology; MQ: metadata quality; NCBI: National Center for Biotechnology Information; NLP: natural language processing; NPFM: NCBI protein family models; PFA: protein function annotation; PNA: potentially new annotation; RAM: random access memory; TC: total coverage; TN: true negative; TP: true positive; TSA: taxon-specific annotation; TSHMM: taxon-specific HMM; TSV: tab-separated value.

Competing Interests

The authors declare that they have no competing interests.

Funding

This research was supported by the Luxembourg National Research Fund PRIDE17/11823097. P.W. acknowledges the European Research Council (ERC-CoG 863664).

Authors' Contributions

Author contributions according to the contributor roles taxonomy (CRediT) were as follows: Conceptualization: P.Q. and P.M.; Data curation: P.Q.; Formal analysis: P.Q.; Funding acquisition: P.W. and P.M.; Investigation: P.Q.; Methodology: P.Q. and P.M.; Project administration: P.Q. and P.M.; Resources: P.Q.; Software: P.Q.; Supervision: P.M. and P.W.; Validation: P.Q. (lead), F.D., and O.H.; Visualization: P.Q.; Writing—original draft: P.Q. (lead), and P.M.; Writing—review and editing: P.Q., P.M., F.D., O.H., and P.W. All authors proofread and approved of the content in the manuscript.

Acknowledgments

The experiments presented in this article were carried out using the high-performance computing facilities of the University of Luxembourg [76]. P.W. acknowledges the European Research Council (ERC-CoG 863664). We thank Tomila Litvishko for proofreading the manuscript. We would like to acknowledge all the creators of the reference data and software used by Mantis—building upon the complementary knowledge of others truly moves the field forward.

References

1. Segata N, Boernigen D, Tickle TL, et al. Computational meta'omics for microbial community studies. Mol Syst Biol 2013;**9**:666.
2. Muller E, Glaab E, May P, et al. Condensing the omics fog of microbial communities. Trends Microbiol 2013;**21**(7): 325–33.
3. Whisstock JC, Lesk AM. Prediction of protein function from protein sequence and structure. Q Rev Biophys 2003;**36**(3):307–40.
4. Arias C, Weisburd B, Stern-Ginossar N, et al. KSHV 2.0: A comprehensive annotation of the Kaposi's sarcoma-associated herpesvirus genome using next-generation sequencing reveals novel genomic and functional features. PLoS Pathog 2014;**10**(1):e1003847.

5. Chapel A, Kieffer-Jaquinod S, Sagné C, et al. An extended proteome map of the lysosomal membrane reveals novel potential transporters. Mol Cell Proteomics 2013;**12**(6):1572–88.
6. Iorizzo M, Senalik DA, Grzebelus D, et al. De novo assembly and characterization of the carrot transcriptome reveals novel genes, new markers, and genetic diversity. BMC Genomics 2011;**12**(1):389.
7. Heintz-Buschart A, May P, Laczny CC, et al. Integrated multi-omics of the human gut microbiome in a case study of familial type 1 diabetes. Nat Microbiol 2016;**2**:16180.
8. Mason OU, Scott NM, Gonzalez A, et al. Metagenomics reveals sediment microbial community response to Deepwater Horizon oil spill. ISME J 2014;**8**(7):1464–75.
9. Pasolli E, Asnicar F, Manara S, et al. Extensive unexplored human microbiome diversity revealed by over 150,000 genomes from metagenomes spanning age, geography, and lifestyle. Cell 2019;**176**(3):649–662.
10. Sureyya Rifaioglu A, Doğan T, Jesus Martin M, et al. DEEPred: Automated protein function prediction with multi-task feed-forward deep neural networks. Sci Rep 2019;**9**(1):7344.
11. Vazquez A, Flammini A, Maritan A, et al. Global protein function prediction from protein-protein interaction networks. Nat Biotechnol 2003;**21**(6):697–700.
12. Borgwardt KM, Ong CS, Schönauer S, et al. Protein function prediction via graph kernels. Bioinformatics 2005;**21**:i47–56.
13. Steinegger M, Meier M, Mirdita M, et al. HH-suite3 for fast remote homology detection and deep protein annotation. BMC Bioinformatics 2019;**20**(1):473.
14. Seemann T. Prokka: rapid prokaryotic genome annotation. Bioinformatics 2014;**30**(14):2068–9.
15. Huerta-Cepas J, Forslund K, Coelho LP, et al. Fast genome-wide functional annotation through orthology assignment by eggNOG-Mapper. Mol Biol Evol 2017;**34**(8):2115–22.
16. Aziz RK, Bartels D, Best AA, et al. The RAST Server: Rapid Annotations using Subsystems Technology. BMC Genomics 2008;**9**(1):75.
17. Ryu JY, Kim HU, Lee SY. Deep learning enables high-quality and high-throughput prediction of enzyme commission numbers. Proc Natl Acad Sci U S A 2019;**116**(28):13996–4001.
18. Zhao B, Hu S, Li X, et al. An efficient method for protein function annotation based on multilayer protein networks. Hum Genomics 2016;**10**(1):33.
19. Szklarczyk D, Gable AL, Lyon D, et al. STRING v11: Protein-protein association networks with increased coverage, supporting functional discovery in genome-wide experimental datasets. Nucleic Acids Res 2019;**47**:D607–13.
20. Deng L, Zhong G, Liu C, et al. MADOKA: An ultra-fast approach for large-scale protein structure similarity searching. BMC Bioinformatics 2019;**20**(19):662.
21. Overbeek R, Begley T, Butler RM, et al. The subsystems approach to genome annotation and its use in the project to annotate 1000 genomes. Nucleic Acids Res 2005;**33**(17):5691–702.
22. Altschul SF, Gish W, Miller W, et al. Basic Local Alignment Search Tool. J Mol Biol 1990;**215**(3):403–10.
23. Buchfink B, Xie C, Huson DH. Fast and sensitive protein alignment using DIAMOND. Nat Methods 2015;**12**(1):59–60.
24. Eddy SR. HMMER; 2020, HMMER: biosequence analysis using profile hidden Markov models.Accessed 22th March 2021, http://hmmer.org.
25. Jones P, Binns D, Chang HY, et al. InterProScan 5: Genome-scale protein function classification. Bioinformatics 2014;**30**(9):1236–40.
26. Lohse M, Nagel A, Herter T, et al. Mercator: A fast and simple web server for genome scale functional annotation of plant sequence data. Plant Cell Environ 2014;**37**(5):1250–8.
27. Wu S, Zhu Z, Fu L, et al. WebMGA: A customizable web server for fast metagenomic sequence analysis. BMC Genomics 2011;**12**:444.
28. Mitchell AL, Almeida A, Beracochea M, et al. MGnify: The microbiome analysis resource in 2020. Nucleic Acids Res 2020;**48**:D570–8.
29. Keegan KP, Glass EM, Meyer F. MG-RAST, a metagenomics service for analysis of microbial community structure and function. Methods Mol Biol 2016;**1399**:207–33.
30. Pfeiffer F, Oesterhelt D. A manual curation strategy to improve genome annotation: Application to a set of haloarchael genomes. Life 2015;**5**(2):1427–44.
31. Yeats C, Redfern OC, Orengo C. A fast and automated solution for accurately resolving protein domain architectures. Bioinformatics 2010;**26**(6):745–51.
32. Ekman D, Bjorklund AK, Frey-Skott J, et al. Multi-domain proteins in the three kingdoms of life: Orphan domains and other unassigned regions. J Mol Biol 2005;**348**(1):231–43.
33. Lees JG, Lee D, Studer RA, et al. Gene3D: Multi-domain annotations for protein sequence and comparative genome analysis. Nucleic Acids Res 2014;**42**:D240–5.
34. Treiber ML, Taft DH, Korf I, et al. Pre- and post-sequencing recommendations for functional annotation of human fecal metagenomes. BMC Bioinformatics 2020;**21**(1):74.
35. Schnoes AM, Brown SD, Dodevski I, et al. Annotation error in public databases: Misannotation of molecular function in enzyme superfamilies. PLoS Comput Biol 2009;**5**(12):e1000605.
36. Friedberg I. Automated protein function prediction—The genomic challenge. Brief Bioinform 2006;**7**(3):225–42.
37. Araujo FA, Barh D, Silva A, et al. GO FEAT: A rapid web-based functional annotation tool for genomic and transcriptomic data. Sci Rep 2018;**8**(1):1794.
38. Klimke W, O'Donovan C, White O, et al. Solving the problem: Genome annotation standards before the data deluge. Stand Genomic Sci 2011;**5**(1):168–93.
39. Standardizing data. Nat Cell Biol 2008;**10**(10):1123–4.
40. Gaikwad SV, Chaugule A, Patil P. Text mining methods and techniques. Intl J Comput Appl 2014;**85**(17):422–5.
41. Wang S, Ma J, Yu MK, et al. Annotating gene sets by mining large literature collections with protein networks. Pac Symp Biocomput 2018;**23**:602–13.
42. Pesquita C, Faria D, Falcão AO, et al. Semantic similarity in biomedical ontologies. PLoS Comput Biol 2009;**5**(7):e1000443.
43. Zeng Z, Shi H, Wu Y, et al. Survey of natural language processing techniques in bioinformatics. Comput Math Methods Med 2015;**2015**:674296.
44. Slater LT, Bradlow W, Ball S, et al. Improved characterisation of clinical text through ontology-based vocabulary expansion. J Biomed Semantics 2020;**12**, doi:10.1186/s13326-021-00241-5.
45. Huang CC, Lu Z. Community challenges in biomedical text mining over 10 years: Success, failure and the future. Brief Bioinform 2016;**17**(1):132–44.
46. Benabderrahmane S, Smail-Tabbone M, Poch O, et al. IntelliGO: A new vector-based semantic similarity measure including annotation origin. BMC Bioinformatics 2010;**11**:588.
47. Peng J, Uygun S, Kim T, et al. Measuring semantic similarities by combining gene ontology annotations and gene co-function networks. BMC Bioinformatics 2015;**16**:44.

48. Liu M, Thomas PD. GO functional similarity clustering depends on similarity measure, clustering method, and annotation completeness. BMC Bioinformatics 2019;**20**(1):155.
49. Daraselia N, Yuryev A, Egorov S, et al. Automatic extraction of gene ontology annotation and its correlation with clusters in protein networks. BMC Bioinformatics 2007;**8**:243.
50. Ehsani R, Drabløs F. TopoICSim: A new semantic similarity measure based on gene ontology. BMC Bioinformatics 2016;**17**(1):296.
51. Kramer M, Dutkowski J, Yu M, et al. Inferring gene ontologies from pairwise similarity data. Bioinformatics 2014;**30**(12):i34–42.
52. Promponas VJ, Iliopoulos I, Ouzounis CA. Annotation inconsistencies beyond sequence similarity-based function prediction – phylogeny and genome structure. Stand Genomic Sci 2015;**10**:108.
53. Ellens KW, Christian N, Singh C, et al. Confronting the catalytic dark matter encoded by sequenced genomes. Nucleic Acids Res 2017;**45**(20):11495–514.
54. Queirós P. Mantis - Wiki. 2020. Acessed 22th March 2021, available at https://github.com/PedroMTQ/mantis/wiki.
55. Aramaki T, Blanc-Mathieu R, Endo H, et al. KofamKOALA: KEGG Ortholog assignment based on profile HMM and adaptive score threshold. Bioinformatics 2020;**36**(7):2251–2.
56. El-Gebali S, Mistry J, Bateman A, et al. The Pfam protein families database in 2019. Nucleic Acids Res 2019;**47**:D427–32.
57. Huerta-Cepas J, Szklarczyk D, Heller D, et al. eggNOG 5.0: A hierarchical, functionally and phylogenetically annotated orthology resource based on 5090 organisms and 2502 viruses. Nucleic Acids Res 2019;**47**:D309–14.
58. Lu S, Wang J, Chitsaz F, et al. CDD/SPARCLE: The conserved domain database in 2020. Nucleic Acids Res 2020;**48**(D1):D265–8.
59. Haft DH, Selengut JD, Richter RA, et al. TIGRFAMs and genome properties in 2013. Nucleic Acids Res 2013;**41**:D387–95.
60. Hyatt D, Chen GL, LoCascio PF, et al. Prodigal: Prokaryotic gene recognition and translation initiation site identification. BMC Bioinformatics 2010;**11**:119.
61. Delogu F, Kunath BJ, Evans PN, et al., Integration of absolute multi-omics reveals dynamic protein-to-RNA ratios and metabolic interplay within mixed-domain microbiomes. Nat Commun 2020;**11**:4708.
62. Kunath BJ, Delogu F, Naas AE, et al. From proteins to polysaccharides: Lifestyle and genetic evolution of *Coprothermobacter proteolyticus*. ISME J 2019;**13**(3):603–17.
63. Busi SB, Pramateftaki P, Brandani J, et al. Optimised biomolecular extraction for metagenomic analysis of microbial biofilms from high-mountain streams. PeerJ 2020;**8**:e9973.
64. Sunagawa S, Coelho LP, Chaffron S, et al. Structure and function of the global ocean microbiome. Science 2015;**348**(6237):1261359.
65. Sigrist CJA, de Castro E, Cerutti L, et al. New and continuing developments at PROSITE. Nucleic Acids Res 2013;**41**: D344–7.
66. Mooney MA, Nigg JT, McWeeney SK, et al. Functional and genomic context in pathway analysis of GWAS data. Trends Genet 2014;**30**(9):390–400.
67. Mavromatis K, Chu K, Ivanova N, et al. Gene context analysis in the integrated microbial genomes (IMG) Data Management System. PLoS One 2009;**4**(11):e7979.
68. Hannigan GD, Prihoda D, Palicka A, et al. A deep learning genome-mining strategy for biosynthetic gene cluster prediction. Nucleic Acids Res 2019;**47**(18):e110.
69. Parks DH, Chuvochina M, Waite DW, et al. A standardized bacterial taxonomy based on genome phylogeny substantially revises the tree of life. Nat Biotechnol 2018;**36**(10):996–1004.
70. Parks DH, Chuvochina M, Chaumeil PA, et al. A complete domain-to-species taxonomy for Bacteria and Archaea. Nat Biotechnol 2020;**38**(9):1079–86.
71. Buell R, Deutschbauer A, Adin D, et al. Breaking the Bottleneck of Genomes: Understanding Gene Function Across Taxa; Workshop Report. United States. 2019. doi:10.2172/1616527
72. Mangul S, Mosqueiro T, Abdill RJ, et al. Challenges and recommendations to improve the installability and archival stability of omics computational tools. PLoS Biol 2019;**17**(6), doi:10.1371/journal.pbio.3000333.
73. Kaur N, Garg D. Analysis of the depth first search algorithms. Data Mining Knowl Eng 2012;**4**:37–41.
74. Behnel S, Bradshaw R, Citro C, et al. Cython: The best of both worlds. Comput Sci Eng 2011;**13**(2):31–9.
75. UniProt Consortium. UniProt: A worldwide hub of protein knowledge. Nucleic Acids Res 2019;**47**:D506–15.
76. Varrette S, Bouvry P, Cartiaux H, et al. Management of an academic HPC cluster: The UL experience. IEEE HPCS; 2014; available at https://hpc.uni.lu.
77. Queirós P, Delogu F, Hickl O, et al. Supporting data for "Mantis: Flexible and consensus-driven genome annotation." GigaScience Database; 2021. http://dx.doi.org/10.5524/100903.