Pedro Queirós*, Polina Novikova, Paul Wilmes and Patrick May

Unification of functional annotation descriptions using text mining

https://doi.org/10.1515/hsz-2021-0125
Received January 27, 2021; accepted May 3, 2021;
published online May 13, 2021

Abstract: A common approach to genome annotation involves the use of homology-based tools for the prediction of the functional role of proteins. The quality of functional annotations is dependent on the reference data used, as such, choosing the appropriate sources is crucial. Unfortunately, no single reference data source can be universally considered the gold standard, thus using multiple references could potentially increase annotation quality and coverage. However, this comes with challenges, particularly due to the introduction of redundant and exclusive annotations. Through text mining it is possible to identify highly similar functional descriptions, thus strengthening the confidence of the final protein functional annotation and providing a redundancy-free output. Here we present UniFunc, a text mining approach that is able to detect similar functional descriptions with high precision. UniFunc was built as a small module and can be independently used or integrated into protein function annotation pipelines. By removing the need to individually analyse and compare annotation results, UniFunc streamlines the complementary use of multiple reference datasets.

Keywords: natural language processing; protein function annotation; text mining; UniFunc.

***Corresponding author: Pedro Queirós**, Systems Ecology, Esch-sur-Alzette, Luxembourg; and Université du Luxembourg, 2, Avenue de l'Université, L-4365 Esch-sur-Alzette, Luxembourg,
E-mail: pedro.queiros@uni.lu. https://orcid.org/0000-0002-0831-4261
Polina Novikova and Paul Wilmes, Systems Ecology, Esch-sur-Alzette, Luxembourg, E-mail: polina.novikova@uni.lu (P. Novikova), paul.wilmes@uni.lu (P. Wilmes). https://orcid.org/0000-0001-7162-2542 (P. Novikova). https://orcid.org/0000-0002-6478-2924 (P. Wilmes)
Patrick May, Bioinformatics Core, Luxembourg Centre for Systems Biomedicine, University of Luxembourg, 4362, Esch-sur-Alzette, Luxembourg, E-mail: patrick.may@uni.lu. https://orcid.org/0000-0001-8698-3770

Introduction

Protein function annotation is the process of identifying regions of interest in a protein sequence and assigning a certain biological function to these regions. It enables the understanding of the physiology and role of single or multiple organisms in a community/ecosystem, which is particularly important to define the metabolic capacities of newly sequenced organisms or communities (Stein 2001). Function assignment is based on reference data, such that if we have an unknown protein X which is similar to protein Y (e.g., by sequence or structure similarity) then we can infer these proteins share the same function(s) (Loewenstein et al. 2009; Whisstock and Lesk 2003). By extent, we can then assume that protein X can be assigned the same protein functional description and/or database identifiers (ID) such as protein Y.

The reference data used for this purpose usually stems from a single source, however, some protein function annotation tools use multiple sources, e.g., InterProScan (Jones et al. 2014). The latter provides several advantages: reinforcement of annotation confidence (several independent sources indicating the same function), improvement of downstream data integration (wider variety of different database IDs), and higher annotation coverage (wider search space). Simultaneously, using multiple references gives rise to various challenges, in particular, uncertainty to define which annotation best represents the functional role of a certain protein (especially when multiple sources infer mutually exclusive functions) and dealing with the introduction of redundancy. These disadvantages are commonly addressed via manual curation, however, this is not feasible in large-scale projects. Many databases provide cross-linking (e.g., UniProt by the UniProt Consortium (2019)), which permits automating this process by checking for intersecting IDs, however, some functional annotations only contain free-text descriptions. Applying the same intersection methodology for text is not viable due to human language's intrinsic richness in confounders (e.g., determiner "the"). This leads to the omission of such functional annotations and may cause the exclusion of potentially useful information. Through the use of text mining techniques, it becomes possible to integrate these functional annotations.

Text mining is the process of exploring and analysing large amounts of unstructured text data aided by software. It allows identifying potential concepts, patterns, topics, keywords, and other attributes in data (Gaikwad et al. 2014). A review by Zeng et al. (2015) has shown some of the potential and diverse techniques and applications of text mining in bioinformatics, some of which include literature mining (Szklarczyk et al. 2019; Wang et al. 2018), protein research (Verspoor et al. 2012), and ontologies (Slater et al. 2021).

We herein present **UniFunc** (Unified Functional annotations), a text mining tool designed to assess the similarity between functional descriptions, thus allowing for the high-throughput integration of data from multiple annotation sources. UniFunc is available at https://github.com/PedroMTQ/UniFunc.

Results

UniFunc

UniFunc's workflow is composed of four main steps: (i) pre-processing, (ii) part-of-speech tagging (PoST), (iii) token encoding and scoring (TES), and (iv) similarity analysis. The first two steps comprise the natural language processing (NLP) of the functional descriptions (e.g., removing extra spaces and eliminating confounders), the last two the text mining (i.e., text encoding and similarity analysis). Figure 1 showcases UniFunc's workflow when comparing two functional descriptions, each step is described in "Material and methods".

While context dependant, we will henceforth refer to an annotation as a functional description of a particular protein (e.g., "glucose degradation"). Each annotation may contain one or multiple sentences, which are composed of one or multiple tokens (e.g., "glucose"). A collection of independent annotations constitutes here the annotation corpus.

Benchmark

As validation, we downloaded all of Swiss-Prot's (UniProt Consortium 2019) protein entries ($N = 563973$, as of 2021/01/16) and selected those that had a functional description, and at least one EC number, Pfam (El-Gebali et al. 2019) ID, or eggNOG (Huerta-Cepas et al. 2018) ID resulting in a validation dataset with 133,450 entries. We then generated two

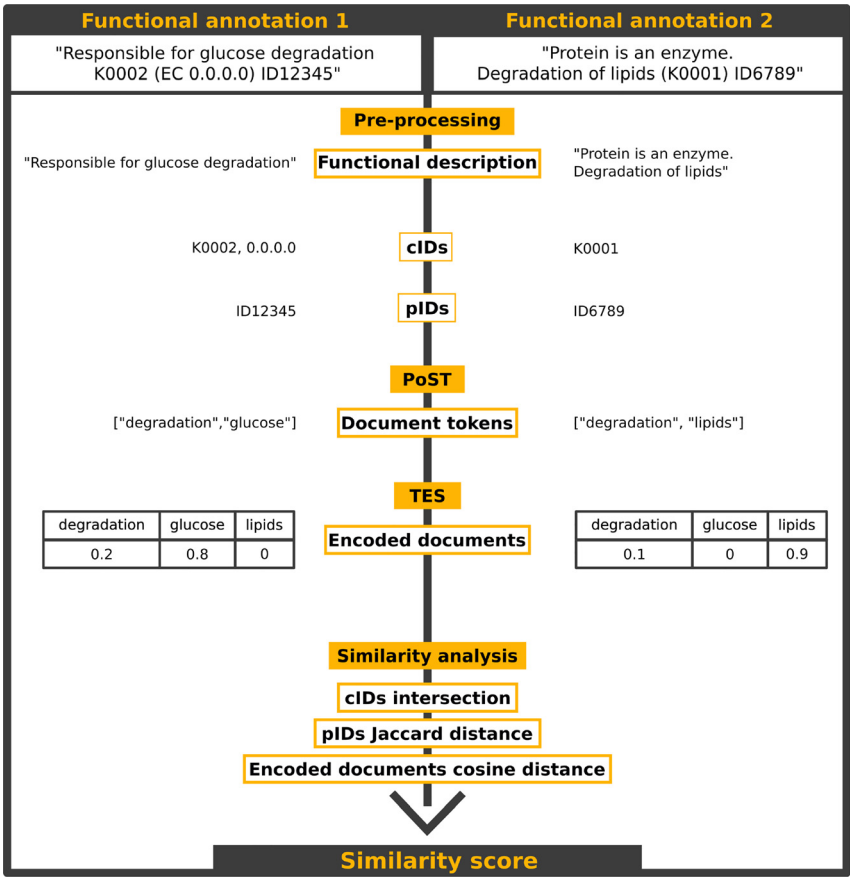

Figure 1: Simplified overview of the UniFunc workflow. UniFunc starts by extracting all the IDs (cIDs and pIDs). It then removes uninformative parts from the annotation and splits it by sentences and tokens. UniFunc then furthers processes the tokens, by tagging each token, and only keeping the most relevant tokens within each annotation and these tokens are encoded into TF-IDF scaled vectors. Finally, the cosine distance between the two annotation vectors and the Jaccard distance between the pIDs are calculated. If any cIDs intersected, the similarity score is 1, otherwise, both previously mentioned distances are used to calculate the entry's similarity score. Abbreviations used in this figure include cIDs (common database identifiers), pIDs (possible database identifiers), PoST (part-of-speech tagging), and TES (token encoding and scoring).

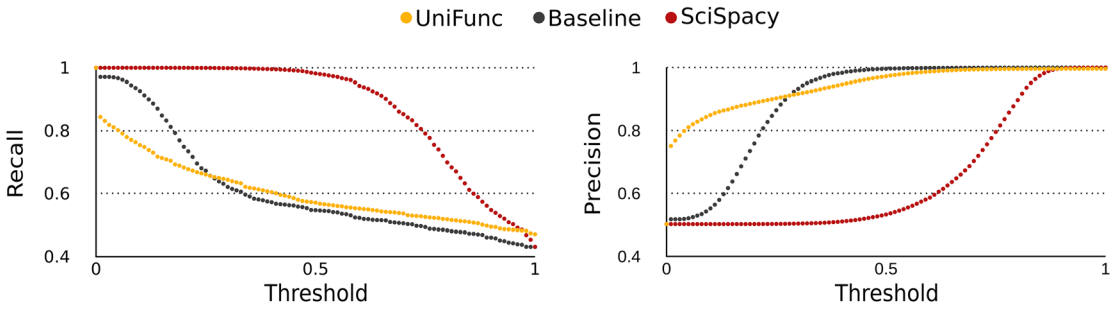

Figure 2: Performance of the baseline and SciSpacy models and UniFunc according to precision and recall.

sets of pairwise functional annotation comparisons. One set of pairs with intersecting identifiers (positive cases) and the other with non-intersecting identifiers (negative case). We then calculated the similarity score of the functional descriptions in each pairwise comparison using different models. In order to understand the impact of the NLP and text mining approach used in UniFunc we built a baseline model that employs very simplistic pre-processing and text encoding methods. We also compared UniFunc against a SciSpacy (Neumann et al. 2019) model, a python library that offers biomedical natural language processing models. With this pre-trained SciSpacy model we used the same simplistic pre-processing as the baseline model. All three models used the same similarity metric, i.e., cosine distance.

The results of this pairwise similarity comparison were then compiled into threshold-specific confusion matrices, where true positives (TP) correspond to pairs of entries with intersecting identifiers and a similarity score above the threshold, true negatives (TN) to non-intersecting identifiers and a similarity score below the threshold, false positives (FP) to non-intersecting identifiers and a similarity score above the threshold, and false negatives (FN) to intersecting identifiers and a similarity score below the threshold. These matrices were then used to calculate several performance metrics: Specificity $= \frac{TN}{TN+FP}$, Precision $= \frac{TP}{TP+FP}$, Recall $= \frac{TP}{TP+FN}$, and $F\beta\text{score} = \frac{(1+\beta^2)\times\text{Precision}\times\text{Recall}}{\beta^2\times\text{Precision}+\text{Recall}}$ with β equal to 1. These metrics are available in the Supplementary excel spreadsheet. We then plotted precision and recall (Figure 2) with a threshold ranging from 0 to 1 in increments of 0.01. The area under the ROC curve (AUC) was also calculated, with the baseline model having an AUC of 0.835, UniFunc 0.868, and SciSpacy 0.876.

Case studies

In the following section, we will provide two case studies that will serve as examples for the potential application of UniFunc. The first entails functionally annotating a sample with multiple reference databases and using UniFunc for integrating these different sources of functional annotations. The second the generation of a Hidden Markov Model (HMM) reference database, using UniFunc to evaluate the functional homogeneity of the HMM. Methodology details are available in the "**Case studies methodology**" section.

Using multiple reference data sources

As an example, we annotated the proteome from the organism *Bacillus subtilis* (Uniprot proteome ID UP000001570) using HMMER (Roberts Eddy 2020) against two reference databases (KOfam by Aramaki et al. 2020 and NCBI's protein family models by Lu et al. 2020).

Using as reference the KOfam HMMs and NCBI's protein family models (NPFM) we annotated 3324 and 2895 out of 4260 sequences, respectively. Combined, both references annotated 3444 sequences. For each sequence, we checked which functional annotations from KOfam and NPFM shared IDs, those that shared IDs were identified as being functionally equal, of which 492 sequences were found. The remaining sequences would then need to be evaluated according to their functional annotation description; doing so manually would only be feasible for a select number of sequences (not feasible in a high-throughput pipeline). As such, UniFunc was used to evaluate the similarity of functional descriptions and thus aid in the integration of these two reference data sources. Using UniFunc, we calculated the similarity between the functional annotation descriptions from KOfam and NPFM. By setting a similarity threshold of 0.9 (merely an example), we found that 266 were in functional agreement. In this manner, we were able to integrate two reference data sources into a single, non-redundant annotation. A similar protocol could be applied for the annotation of metagenome-assembled genomes or full metagenomes.

Functional homogeneity of a protein cluster

During the creation of multiple sequence alignment-based reference data sources, e.g., HMMs, it is common to cluster protein sequences by their sequence and functional similarity. UniFunc can automate functional similarity analysis. As an example, we evaluated the functional homogeneity and average pairwise sequence distance of a collection ($N = 4100$) of clustered *Archaea* protein sequences. We then kept only the clusters with at least 10 proteins sequences and with a minimum 80% of functionally annotated protein sequences, resulting in 2516 clusters. We then used UniFunc to measure the functional pairwise similarity (i.e., cluster functional homogeneity) between each pair of protein sequences within the clusters. These clusters were further refined by setting a minimum of 0.9 functional homogeneity and a maximum average pairwise sequence distance of 0.1, thus obtaining 2182 highly homogeneous clusters (function and sequence-wise). These new clusters could then be used to create highly-specific HMMs, which could then be used to functionally annotate *Archaea* samples.

Discussion

We have developed a method that processes and encodes free-text functional descriptions, allowing for high-throughput pairwise similarity analysis, ultimately enabling the comparison of functional annotations obtained from multiple sources. We designed UniFunc with two applications in mind, first to facilitate redundancy elimination when consolidating protein function annotations, secondly as a cross-linking mechanism for functional annotations devoid of IDs. While more sophisticated methodologies have been applied in related fields (Lee et al. 2020; Neumann et al. 2019; Weber et al. 2021), we aimed to create a method with low complexity that could be easily integrated into more complex annotation pipelines.

UniFunc was developed to identify and eliminate confounders (e.g., determiner "the") from functional annotations (i.e., noise reduction). Without noise reduction, annotations such as "this is an oxidase" and "this is a kinase" may be indicated as being similar (three out of five identical tokens). On the other hand, without noise reduction, annotations such as "believed to be an oxidase" and "this is an oxidase" may be indicated as being dissimilar (two out of seven identical tokens). Fundamentally, confounder elimination increases "purity" of annotations, resulting in more polarized (close to 0 or 1) similarity scores.

As seen in Figure 2 UniFunc can achieve high precision, in fact, initial convergence to high precision was much faster for UniFunc than for the other models. This is explained by the extreme distribution of the similarity scores, which registers high density toward 0 or 1, whilst in the other models they are more evenly distributed. This extreme distribution of the scores confirms UniFunc's superior noise reduction ability, which is especially useful when comparing functional descriptions with a high amount of confounders (e.g., when comparing functional annotations with multiple sentences).

Neither UniFunc nor the baseline model achieved high recall at higher similarity thresholds. This can be explained by the fact that while two functional annotations may imply the same function (and thus share IDs), they may also use different nomenclature (e.g., quercetin can also be called meletin or xanthaurine). In these scenarios, a comprehensive biological lexicon would be required, and while such lexicons exist (Thompson et al. 2011), they are usually behind pay-walls, which limits their use by open source projects following FAIR principles set by Wilkinson et al. (2016). Since several databases (e.g., Caspi et al. 2019; Gene Ontology Consortium 2019) provide ontology systems, a future solution may be to use these to create a lexicon from these. Improving UniFunc's NLP (e.g., lemmatization – "methyltransferase" to "methyl transfer enzyme"), could aid in the reduction of false negatives. Among various text mining techniques, word embedding (Mikolov et al. 2013; Pennington et al. 2014) could prove beneficial; this technique permits identifying tokens as similar when they appear in similar contexts. This could prove advantageous in UniFunc's workflow (i.e., specifically during part-of-speech tagging), where instead of removing certain lexical tags (e.g., pronouns), we could use word embedding to understand how related two tokens are (e.g., "enzyme" is closer to "protein" than to "gene"). Word embedding, or similar techniques, could potentially improve UniFunc's recall. However, this added complexity could also reduce UniFunc's robustness when dealing with tokens absent or underrepresented in the annotation corpus (e.g., some tokens appear only once in the whole corpus). In addition, higher complexity techniques also tend to be more costly, both in term of time and hardware requirements.

At higher thresholds, UniFunc had a higher recall than the baseline model, while having similar precision. This is again due to noise reduction and its resulting extreme distribution of scores. Since the baseline model does not eliminate confounders, its ability to produce high similarity scores is reduced, leading to a lower recall when the similarity threshold is too high. SciSpacy and UniFunc

behaved very differently along the different thresholds. UniFunc quickly achieved high precision at the expense of a rapid decrease in recall, whereas SciSpacy only achieved a higher precision at higher thresholds. Despite this, the AUC difference between the two model was low, UniFunc's AUC was 0.008 lower than SciSpacy's, in essence, both have their own advantages. SciSpacy is applicable on a broader range of scenarios, whereas UniFunc was specifically implemented to compare protein function annotations. On the other hand, we believe that UniFunc's implementation may offer more future-proofing as its corpus can be updated by re-downloading Swiss-Prot's protein annotations and gene ontologies.

Overall, all models had an AUC above 0.8, indicating that these performed well in predicting the correct classes of the validation dataset. At a baseline, we consider UniFunc's performance to be consistent (high precision) to allow for its reliable use in the scenarios it was built for. Future iterations will address UniFunc's lower recall.

We have also provided two case studies where UniFunc could be easily applied. In the first, we have shown that UniFunc can aid the integration of multiple reference data sources. This methodology could also prove advantageous when functionally annotating samples with very specific reference data sources (e.g., Resfams Gibson et al. 2015). In addition, we have shown that UniFunc can be used to measure functional homogeneity when creating functional reference data sources.

In conclusion, while the standardization of functional annotation data (including IDs and free-text) still constitutes a challenge for successful data integration (Huang and Lu 2016; Lapatas et al. 2015; Stein 2003), we have shown that functional descriptions from multiple references/sources can be successfully integrated using text mining, complementing ID-based data integration.

Materials and methods

Baseline model workflow

In the baseline model, descriptions are split into tokens by using spaces as delimiters. Each annotation is then encoded according to the presence (1) or absence (0) of all tokens within both annotations (i.e., one-hot encoding). For example, if the first annotation contains the tokens ("lipids", "degradation") and the second ("glucose", "degradation"), the union of tokens would correspond to ("lipids", "glucose", "degradation"). We then check the presence/absence of the first token "lipids" in the first annotation: since it is present, we add a "1" to the first annotation vector. We do the same for the second annotation: since it is absent, we add "0" to the second annotation vector. We repeat the same process for the second token "glucose", so the first annotation vector now is [1, 0] and the second [0, 1]. We then do the same for all the remaining tokens (i.e., "degradation") and obtain, for each annotation, a vector with the number of token as entries. Here, where the first annotation is encoded as [1, 0, 1] and the second as [0, 1, 1]. One minus the cosine distance of these vectors will correspond to the similarity of these two annotations.

UniFunc workflow

UniFunc's workflow is comprised of four main steps: (i) pre-processing, (ii) part-of-speech tagging, (iii) token encoding and scoring, and (iv) similarity analysis. NLP includes steps i and ii, which were tailored towards the removal of confounder tokens, thus preserving only the most significant tokens within the functional description. Steps iii and iv refer to the text mining part of the workflow and involves the encoding of annotations, making them comparable. Figure 1 shows an overview of UniFunc workflow.

Pre-processing: NLP starts with ID extraction via the use of regular expressions in two manners, the first looks for common database IDs (cID) patterns (i.e., enzyme ECs – Cornish-Bowden (2014); TCDB – Saier et al. (2006), KO – Kanehisa and Goto (2000); TIGRfam – Haft et al. (2013); Pfam – El-Gebali et al. (2019); COG – Tatusov et al. (2000); and GO – Gene Ontology Consortium (2019)), the second finds possible IDs (pID) with the regular expression "`[A-Z]+\d{3,}(\.\d+)?([A-Z]+)?`", which captures "words" with capital letters followed by a set of numbers (possibly followed by a dot and digits or more capital letters). While ID structure varies, in our experience, this is the most standardized format across multiple databases. Protein acronyms are also captured by this regular expression, which, due to the increased weight of pIDs in comparison to tokens, will increase the similarity score of annotations containing the same protein acronyms. IDs are put aside for later use during similarity analysis.

After ID extraction, the annotation is pre-processed, where unnecessary punctuation and filler entities (e.g., extra spaces) are removed, followed by standardization of nomenclature (e.g., "–>" to "to") and numerals (e.g., "III" to "3"). Annotations are then split into sentences (sentence segmentation) and each sentence is split into tokens (tokenization, e.g., "lipid degradation" is split into two tokens "lipid" and "degradation"). Each plural token is then converted to its singular form (stemming). Finally, we also aggregate certain tokens (e.g., tokens "terminal" and "N" to token "terminal N") into a single token. Tokens within parentheses are removed (except when they are acronyms) as, in our experience, they tend to contain tokens irrelevant for similarity analysis (e.g., "Seems to play a role in the dimerization of PSII [By similarity].").

Part-of-speech tagging: Part-of-speech tagging (PoST) is the method of lexically classifying/tagging tokens based on their definition and context in the sentence. The aim here is the identification and elimination of tokens that could introduce noise during similarity analysis (e.g., the very common determiner "the"). We use two taggers, a custom tagger, and NLTK's pre-trained Perceptron tagger (Bird et al. 2009; Honnibal 2013). The first tagger is independent of context and uses Wordnet's lexicon (Miller 1995) to identify the most common lexical category of any given token. Should a token be present in Wordnet's lexicon, a list of potential lexical categories for the token is given (e.g., noun, synonym, verb, etc.), the token is then assigned the most common tag. To adjust this tagger's lexicon to biological data,

gene ontologies (Ashburner et al. 2000; Gene Ontology Consortium 2019) names, synonyms, and definitions are processed and tagged (using the pre-trained Perceptron tagger). Those not classified as adpositions, conjunctions, determinants, pronouns, or particles are added to the custom tagger as nouns. Tokens are then classified by both taggers, tokens untagged by the custom Wordnet tagger are assigned the Perceptron's tag. The Perceptron tagger is only used as a backup since it has been pre-trained with the Penn Treebank dataset (Taylor et al. 2003), thus its corpus is unspecific to UniFunc's target data. Finally, tokens that have not been tagged as adjectives, adverbs, nouns, or verbs, that belong to a pre-compiled list of common biological terms, or are common English stop words are removed from the annotation. Ideally, more tag types would be removed (e.g., adverbs), however, we found that doing so eliminated important tokens. In addition, since some annotations may contain synonym tokens (e.g., an annotation contains the token "identical" and another annotation contains the token "equal"), we use Wordnet to find synonyms and replace the respective tokens, such that both annotations contain the same token. This is only applicable to the tokens kept after PoST.

Token encoding and scoring: Token encoding is similar to the baseline's model token encoding, with a key difference, unlike the baseline model's binary vector, UniFunc uses a Term Frequency-Inverse Document Frequency (*TF-IDF*) scaled vector. Therefore, the annotation vectors will contain only elements with values ranging from 0 to 1, the higher the value the more important the token is in the annotation. TF-IDF measures the importance of each token relative to an annotation and the annotation corpus, therefore tokens that are frequent in a single annotation but infrequent in the annotation corpus receive a higher weight. *TF-IDF* has been successfully used in past projects, such as Benabderrahmane et al. (2010) and Huang et al. (2012). As a reference corpus, we downloaded all of Swiss-Prot's protein entries (N = 563973, as of 2021/01/16) and their respective functional description ("Function [CC]") and protein names, as well as the gene ontologies "go.obo" file. From the go.obo file we extracted the "name:", "synonym:", and "def:" entries, from the Swiss-Prot file, all the functional descriptions and protein names. This data was then pre-processed (using the same method used by UniFunc) and split into tokens, we then created a frequency table with the number of times each token appeared in the corpus.

TF-IDF is calculated with the equation $\frac{NW}{TW} \times \frac{TC}{NC}$, where NW is the number of times a token appears in the annotation, TW the total number of tokens in the annotation, TC the total number of annotation in the annotation corpus, and NC the total number of times a certain token appears in the corpus. We apply a log10 scale to reduce the distance between the vectors' elements and a MinMax scale to sort the tokens by their intra-annotation importance.

Similarity analysis: Each functional description now has an associated set of IDs and an annotation vector. When two functional descriptions share a cID (i.e., enzyme ECs, TCDB, KO, TIGRfam, Pfam, COG, and GO), the similarity score corresponds to 1. When both functional descriptions have pIDs we calculate the Jaccard distance between these sets of pIDs and subtract it to 1, obtaining pIDs similarity pID_{sim}. We then calculate the cosine distance between both annotations and subtract it to 1 to obtain the annotation similarity Doc_{sim}. If pID_{sim} is above 0, then the similarity score corresponds to $\frac{2 \times pID_{sim} + Doc_{sim}}{3}$, otherwise, it corresponds to Doc_{sim}.

SciSpacy models workflow

As an additional comparison we used on SciSpacy's (Neumann et al. 2019) "en_core_sci_lg" model. SciSpacy offers biomedical natural language processing models, which are integrated into the Spacy (Honnibal et al. 2020) framework, a natural language processing Python library. In this model, descriptions are split into tokens by using spaces as delimiters (similarly to the baseline model).

Case studies methodology

For the "**Using multiple reference data sources**" case study the functional annotations were generated by using HMMER's *hmmsearch* command against the KOfam and NPFM HMMs. Since NPFM provides taxon-specific HMMs, protein sequences were annotated hierarchically, meaning that, if available, we used the HMMs respective to each taxon of the *B. subtilis* taxonomic lineage (i.e., 131567> 2 > 1783272 > 1239 > 91061 > 1385 > 186817 > 1386 > 653685 > 1423). In each *hmmsearch* iteration, only the protein sequences left to annotate were used. The functional annotations metadata was then assigned to the hits from HMMER's output.

For the "**Functional homogeneity of a protein cluster**" case study, a collection of 4570 archaeal genomes was downloaded from different sources, 1162 from the UHGG collection of MGnify (Mitchell et al. 2019), 371 from NCBI (Coordinators 2017), and 3037 from the GEM catalogue (Nordberg et al. 2013). CheckM (Parks et al. 2015) was run on the entire collection to ensure high-quality archaeomes (>50% completeness, <5% contamination). Sourmash (Brown and Irber 2016) was used to identify groups of highly similar genomes in the collection. Similarity of genomes in each group was validated based on their GC content and related taxonomy. Within each group of highly similar genomes, a genome with the highest completeness and lowest contamination was selected as a group representative. As a result, a collection of 1681 non-redundant high-quality archaeal genomes was composed and used to create archaea-specific HMMs. Protein-coding genes were predicted with Prodigal (Hyatt et al. 2010), and functionally annotated with Mantis (Queirós et al. 2020). MMseqs2 (Steinegger and Söding 2017) was used to cluster proteins by sequence similarity. Average pairwise distance of each cluster was calculated with Clustal omega (Sievers et al. 2011). Protein clusters were used to build multiple sequence alignments (MSAs) using MUSCLE (Edgar 2004), and the MSAs were used to construct HMMs using HMMER (Roberts Eddy 2020).

Model validation

In order to understand the performance impact of the NLP (context-specific pre-processing and PoST) and encoding (TF-IDF scaled document encoding) approach used by UniFunc, we compared its performance against the previously described baseline model.

As a validation dataset, we started by downloading all the Swiss-Prot (UniProt Consortium 2019) entries (N = 563973, as of 2021/01/16) with the columns "Entry", "Function [CC]", "EC number","Cross-reference (Pfam)", and "Cross-reference (eggNOG)". All the entries with a functional description ("Function [CC]"), and at least one EC number, Pfam ID, and eggNOG ID, were selected, resulting in a validation dataset with a total of 133,450 entries.

This dataset allows for the benchmark of each model's ability to correctly identify similar and non-similar functional descriptions,

where similar functional descriptions should have intersecting identifiers (positive class), and non-similar descriptions non-intersecting identifiers (negative class).

Each entry (with a set of IDs S1 and a description D1) in the validation dataset is paired with up to 500 other entries (with a set of IDs S2 and description D2) where $S1 \cap S2 \neq \emptyset$ (positive cases). We then randomly selected an equal number of pairs where $S1 \cap S2 = \emptyset$ (negative cases). As an example, a positive case for the entry "glucose degradation K01" (S1 = {K01}) would be "enzyme that uses glucose as a substrate K01 KO2" (S2 = {K01, K02}), whereas a negative case for the same entry would be "lipid degradation KO3" (S2 = {K03}). In the positive case the ID KO1 is common to both entries ($\{KO1\} \cap \{KO1, KO2\} \neq \emptyset$), whereas in the negative case no IDs are shared ($\{KO1\} \cap \{KO3\} = \emptyset$). Assigning an equal number of positive and negative cases to each entry ensures class balance which validates AUC as a global performance metric (Jeni et al. 2013).

We then use UniFunc and the other models to calculate the similarity score (SS) between each pair of entries' description. Finally, confusion matrices are created with threshold $(T) \in [0, 1, 0.1]$, where true positives $(TP) = S1 \cap S2 \neq \emptyset \wedge SS \geqslant T$, true negatives $(TN) = S1 \cap S2 = \emptyset \wedge SS < T$, false positives $(FP) = S1 \cap S2 = \emptyset \wedge SS \geqslant T$, and false negatives $(FN) = S1 \cap S2 \neq \emptyset \wedge SS < T$. The test case entries are the same for all models. During benchmark, UniFunc does not use IDs for similarity analysis.

Acknowledgements: The experiments presented in this paper were carried out using the HPC facilities of the University of Luxembourg (Varrette et al. 2014).

Author contributions: Author contributions according to the contributor roles taxonomy CRediT was as follows: Conceptualization: P.Q.; Data curation: P.Q.; Formal Analysis: P.Q.; Funding acquisition: P.W. and P.M.; Investigation: P.Q.; Methodology: P.Q.; Project administration: P.Q. and P.M.; Resources: P.Q. and P.N.; Software: P.Q.; Supervision: P.M. and P.W.; Validation: P.Q.; Visualization: P.Q.; Writing – original draft: P.Q. (lead), and P.M.; Writing – review & editing: P.Q., P.N, P.M., and P.W. All authors proof-read and approved of the content in this research paper.

Research funding: This study was supported by the Luxembourg National Research Fund PRIDE17/11823097.

Conflict of interest statement: The authors declare that they have no competing interests.

References

Aramaki, T., Blanc-Mathieu, R., Endo, H., Ohkubo, K., Kanehisa, M., Goto, S., and Ogata, H. (2020). KofamKOALA: KEGG oOrtholog assignment based on profile HMM and adaptive score threshold. Bioinformatics 36: 2251–2252.

Ashburner, M., Ball, C.A., Blake, J.A., Botstein, D., Butler, H., Cherry, J.M., Davis, A.P., Dolinski, K., Dwight, S.S., Eppig, J.T., et al. (2000). Gene ontology: tool for the unification of biology. Nat. Genet. 25: 25–29.

Benabderrahmane, S., Smail-Tabbone, M., Poch, O., Napoli, A., and Devignes, M.-D. (2010). IntelliGO: a new vector-based semantic similarity measure including annotation origin. BMC Bioinf. 11, https://doi.org/10.1186/1471-2105-11-588.

Bird, S., Klein, E., and Loper, E. (2009). Natural language processing with python, Available at: <https://www.nltk.org/book/>.

Brown, C.T. and Irber, L. (2016). sourmash: a library for MinHash sketching of DNA. J. Open Source Softw. 1: 27.

Caspi, R., Billington, R., Keseler, I.M., Kothari, A., Krummenacker, M., Midford, P.E., Ong, W.K., Paley, S., Subhraveti, P., and Karp, P.D. (2019). The MetaCyc database of metabolic pathways and enzymes – a 2019 update. Nucleic Acids Res. 48: 445–453.

Coordinators, N.R. (2017). Database resources of the national center for biotechnology information. Nucleic Acids Res. 46: 8–13.

Cornish-Bowden, A. (2014). Current IUBMB recommendations on enzyme nomenclature and kinetics. Perspect. Sci. 1: 74–87.

Edgar, R.C. (2004). MUSCLE: a multiple sequence alignment method with reduced time and space complexity. BMC Bioinf. 5: 1–19.

El-Gebali, S., Mistry, J., Bateman, A., Eddy, S.R., Luciani, A., Potter, S.C., Qureshi, M., Richardson, L.J., Salazar, G.A., Smart, A., et al. (2019). The Pfam protein families database in 2019. Nucleic Acids Res. 47: 427–432.

Gaikwad, S., Chaugule, A., and Patil, P. (2014). Text mining methods and techniques. Int. J. Comput. Appl. 85: 42–45.

Gene Ontology Consortium (2019). The gene ontology resource: 20 years and still going strong. Nucleic Acids Res. 47: 330–338.

Gibson, M.K., Forsberg, K.J., and Dantas, G. (2015). Improved annotation of antibiotic resistance determinants reveals microbial resistomes cluster by ecology. ISME J. 9: 207–216.

Haft, D.H., Selengut, J.D., Richter, R.A., Harkins, D., Basu, M.K., and Beck, E. (2013). TIGRFAMs and genome properties in 2013. Nucleic Acids Res. 41: 387–395.

Honnibal, M. (2013). A good part-of-speech tagger in about 200 lines of python, Available at: <https://explosion.ai/blog/part-of-speech-pos-tagger-in-python>.

Honnibal, M., Montani, I., Van Landeghem, S., and Boyd, A. (2020). spaCy: industrial-strength natural language processing in python, https://doi.org/10.5281/zenodo.1212303.

Huang, C.-C. and Lu, Z. (2016). Community challenges in biomedical text mining over 10 years: success, failure and the future. Briefings Bioinf. 17: 132–144.

Huang, Y., Gan, M., and Jiang, R. (2012). Ontology-based genes similarity calculation with TF-IDF. LNCS 7473: 600–607.

Huerta-Cepas, J., Szklarczyk, D., Heller, D., Hernández-Plaza, A., Forslund, S.K., Cook, H., Mende, D.R., Letunic, I., Rattei, T., Jensen, L., et al. (2018). eggNOG 5.0: a hierarchical, functionally and phylogenetically annotated orthology resource based on 5090 organisms and 2502 viruses. Nucleic Acids Res. 47: 309–314.

Hyatt, D., Chen, G.-L., LoCascio, P.F., Land, M.L., Larimer, F.W., and Hauser, L.J. (2010). Prodigal: prokaryotic gene recognition and translation initiation site identification. BMC Bioinf. 11, https://doi.org/10.1186/1471-2105-11-119.

Jeni, L.A., Cohn, J.F., and De La Torre, F. (2013). Facing imbalanced data–recommendations for the use of performance metrics. In: *2013 Humaine association conference on affective computing and intelligent interaction*. IEEE, Geneva, Switzerland, pp. 245–251.

Jones, P., Binns, D., Chang, H.-Y., Fraser, M., Li, W., McAnulla, C., McWilliam, H., Maslen, J., Mitchell, A., Nuka, G., et al. (2014).

InterProScan 5: genome-scale protein function classification. Bioinformatics 30: 1236–1240.

Kanehisa, M. and Goto, S. (2000). KEGG: kyoto encyclopedia of genes and genomes. Nucleic Acids Res. 28: 27–30.

Lapatas, V., Stefanidakis, M., Jimenez, R.C., Via, A., and Schneider, M.V. (2015). Data integration in biological research: an overview. J. Biol. Res. (Thessalon) 22, https://doi.org/10.1186/s40709-015-0032-5.

Lee, J., Yoon, W., Kim, S., Kim, D., Kim, S., So, C.H., and Kang, J. (2020). BioBERT: a pre-trained biomedical language representation model for biomedical text mining. Bioinformatics 36: 1234–1240.

Loewenstein, Y., Raimondo, D., Redfern, O.C., Watson, J., Frishman, D., Linial, M., Orengo, C., Thornton, J., and Tramontano, A. (2009). Protein function annotation by homology-based inference. Genome Biol. 10: 1–8.

Lu, S., Wang, J., Chitsaz, F., Derbyshire, M.K., Geer, R.C., Gonzales, N.R., Gwadz, M., Hurwitz, D.I., Marchler, G.H., Song, J.S., et al. (2020). CDD/SPARCLE: the conserved domain database in 2020. Nucleic Acids Res. 48: 265–268.

Mikolov, T., Chen, K., Corrado, G., and Dean, J. (2013). Efficient estimation of word representations in vector space. arXiv preprint, https://arxiv.org/abs/1301.3781.

Miller, G.A. (1995). WordNet: a lexical database for English. Commun. ACM 38: 39–41.

Mitchell, A.L., Almeida, A., Beracochea, M., Boland, M., Burgin, J., Cochrane, G., Crusoe, M.R., Kale, V., Potter, S.C., and Richardson, L.J., et al. (2019). MGnify: the microbiome analysis resource in 2020. Nucleic Acids Res. 48: 570–578.

Neumann, M., King, D., Beltagy, I., and Ammar, W. (2019). *ScispaCy: fast and robust models for biomedical natural language processing*. ACL, Stroudsburg, USA, pp. 319–327.

Nordberg, H., Cantor, M., Dusheyko, S., Hua, S., Poliakov, A., Shabalov, I., Smirnova, T., Grigoriev, I.V., and Dubchak, I. (2013). The genome portal of the department of energy joint genome institute: 2014 updates. Nucleic Acids Res. 42: 26–31.

Parks, D.H., Imelfort, M., Skennerton, C.T., Hugenholtz, P., and Tyson, G.W. (2015). CheckM: assessing the quality of microbial genomes recovered from isolates, single cells, and metagenomes. Genome Res. 25: 1043–1055.

Pennington, J., Socher, R., and Manning, C.D. (2014). *GloVe: global vectors for word representation*. ACL, Stroudsburg, USA, pp. 1532–1543.

Queirós, P., Delogu, F., Hickl, O., May, P., and Wilmes, P. (2020). Mantis: exible and consensus-driven genome annotation. bioRxiv, https://doi.org/10.1101/2020.11.02.360933.

Roberts Eddy, S. (2020). HMMER: biosequence analysis using profile hidden Markov models, Available at: <http://hmmer.org/>.

Saier Milton, H, J., Tran, C.V., and Barabote, R.D. (2006). TCDB: the transporter classification database for membrane transport protein analyses and information. Nucleic Acids Res. 34: 181–186.

Sievers, F., Wilm, A., Dineen, D., Gibson, T.J., Karplus, K., Li, W., Lopez, R., McWilliam, H., Remmert, M., Soding, J., et al. (2011). Fast, scalable generation of high-quality protein multiple sequence alignments using Clustal Omega. Mol. Syst. Biol. 7, https://doi.org/10.1038/msb.2011.75.

Slater, L.T., Bradlow, W., Ball, S., Hoehndorf, R., and Gkoutos, G.V. (2021). Improved characterisation of clinical text through ontology-based vocabulary expansion. J. Biomed. Semant. 12, doi:10.1186/s13326-021-00241-5.

Stein, L. (2001). Genome annotation: from sequence to biology. Nat. Rev. Genet. 2: 493–503.

Stein, L.D. (2003). Integrating biological databases. Nat. Rev. Genet. 4: 337–345.

Steinegger, M., and Söding, J. (2017). MMseqs2 enables sensitive protein sequence searching for the analysis of massive data sets. Nat. Biotechnol. 35: 1026–1028.

Szklarczyk, D., Gable, A.L., Lyon, D., Junge, A., Wyder, S., Huerta-Cepas, J., Simonovic, M., Doncheva, N.T., Morris, J.H., and Bork, P., et al. (2019). STRING v11: protein-protein association networks with increased coverage, supporting functional discovery in genome-wide experimental datasets. Nucleic Acids Res. 47: 607–613.

Tatusov, R.L., Galperin, M.Y., Natale, D.A., and Koonin, E.V. (2000). The COG database: a tool for genome-scale analysis of protein functions and evolution. Nucleic Acids Res. 28: 33–36.

Taylor, A., Marcus, M., and Santorini, B. (2003). *The Penn treebank: an overview*. In: Abeillé, A. (Ed.). Springer, Netherlands, pp. 5–22, https://doi.org/10.1007/978-94-010-0201-1_1.

Thompson, P., McNaught, J., Montemagni, S., Calzolari, N., del Gratta, R., Lee, V., Marchi, S., Monachini, M., Pezik, P., and Quochi, V., et al. (2011). The BioLexicon: a large-scale terminological resource for biomedical text mining. BMC Bioinf. 12, https://doi.org/10.1186/1471-2105-12-397.

UniProt Consortium (2019). UniProt: a worldwide hub of protein knowledge. Nucleic Acids Res. 47: 506–515.

Varrette, S., Bouvry, P., Cartiaux, H., and Georgatos, F. (2014). Management of an academic HPC cluster: the UL experience, Available at: <https://hpc.uni.lu>.

Verspoor, K.M., Cohn, J.D., Ravikumar, K.E., and Wall, M.E. (2012). Text mining improves prediction of protein functional sites. PloS One 7: 1–16.

Wang, S., Ma, J., Yu, M.K., Zheng, F., Huang, E.W., Han, J., Peng, J., and Ideker, T. (2018). Annotating gene sets by mining large literature collections with protein networks. Pac. Symp. Biocomput. 23: 602–613.

Weber, L., Sänger, M., Münchmeyer, J., Habibi, M., Leser, U., and Akbik, A. (2021). HunFlair: an easy-to-use tool for state-of-the-art biomedical named entity recognition. Bioinformatics, https://doi.org/10.1093/bioinformatics/btab042.

Whisstock, J.C. and Lesk, A.M. (2003). Prediction of protein function from protein sequence and structure. Q. Rev. Biophys. 36: 307–340.

Wilkinson, M.D., Dumontier, M., Aalbersberg, I.J., Appleton, G., Axton, M., Baak, A., Blomberg, N., Boiten, J.-W., Silva Santos, L.B.da, Bourne, P.E., et al. (2016). The FAIR guiding principles for scientific data management and stewardship. Sci. Data 3, https://doi.org/10.1038/sdata.2016.18.

Zeng, Z., Shi, H., Wu, Y., and Hong, Z. (2015). Survey of natural language processing techniques in bioinformatics. Comput. Math. Methods Med. 2015, https://doi.org/10.1155/2015/674296.

Supplementary Material: The online version of this article offers supplementary material (https://doi.org/10.1515/hsz-2021-0125).