BMC Bioinformatics

RESEARCH **Open Access**

Comparative Assessment of Protein Large Language Models for Enzyme Commission Number Prediction

João Capela[1*], Maria Zimmermann-Kogadeeva[2], Aalt D. J. van Dijk[3,4], Dick de Ridder[3], Oscar Dias[1,5] and Miguel Rocha[1,5]

*Correspondence:
joao.capela@ceb.uminho.pt

[1] Centre of Biological Engineering, University of Minho, Braga 4710-057, Portugal
[2] Genome Biology Unit, European Molecular Biology Laboratory, Heidelberg, Germany
[3] Bioinformatics Group, Department of Plant Sciences, Wageningen University and Research, Wageningen, The Netherlands
[4] Biosystems Data Analysis, University of Amsterdam, Amsterdam, The Netherlands
[5] LABBELS - Associate Laboratory, Braga/Guimarães, Portugal

Abstract

Background: Protein large language models (LLM) have been used to extract representations of enzyme sequences to predict their function, which is encoded by enzyme commission (EC) numbers. However, a comprehensive comparison of different LLMs for this task is still lacking, leaving questions about their relative performance. Moreover, protein sequence alignments (e.g. BLASTp or DIAMOND) are often combined with machine learning models to assign EC numbers from homologous enzymes, thus compensating for the shortcomings of these models' predictions. In this context, LLMs and sequence alignment methods have not been extensively compared as individual predictors, raising unaddressed questions about LLMs' performance and limitations relative to the alignment methods. In this study, we set out to assess the performance of ESM2, ESM1b, and ProtBERT language models in their ability to predict EC numbers, comparing them with BLASTp, against each other and against models that rely on one-hot encodings of amino acid sequences.

Results: Our findings reveal that combining these LLMs with fully connected neural networks surpasses the performance of deep learning models that rely on one-hot encodings. Moreover, although BLASTp provided marginally better results overall, DL models provide results that complement BLASTp's, revealing that LLMs better predict certain EC numbers while BLASTp excels in predicting others. The ESM2 stood out as the best model among the LLMs tested, providing more accurate predictions on difficult annotation tasks and for enzymes without homologs.

Conclusions: Crucially, this study demonstrates that LLMs still have to be improved to become the gold standard tool over BLASTp in mainstream enzyme annotation routines. On the other hand, LLMs can provide good predictions for more difficult-to-annotate enzymes, particularly when the identity between the query sequence and the reference database falls below 25%. Our results reinforce the claim that BLASTp and LLM models complement each other and can be more effective when used together.

Keywords: Enzyme annotation, Deep learning, Large language models

Background

Enzymes are natural catalysts of biochemical reactions, accelerating reaction rates by decreasing the activation energy required to form or break chemical bonds [1]. Enzymes generally demonstrate specificity for particular substrates, although they can be promiscuous towards structurally similar compounds [2]. Accurately and efficiently predicting their function is important for enhancing genome annotation workflows [3]. These predictions are critical for establishing connections between proteins and reactions, devising new pathways, and gaining precise insights into cellular metabolism.

Enzyme Commission (EC) numbers offer a hierarchical framework, organising enzyme functions. In the context of genome annotation procedures, EC numbers are assigned to protein sequences to succinctly encapsulate patterns of chemical reactions [4]. These correspond to the chemical changes facilitated by the enzyme linked to the respective EC number.

Given the low proportion of enzymes assigned with EC numbers [5], many tools have emerged for this annotation task, leveraging similarity and machine learning (ML) techniques. The former is based on the widely held idea that enzymes sharing high sequence similarity will likely have similar functions [6–8]. This annotation transfer technique is commonly adopted using the gold standard tool BLASTp (Basic Local Alignment Search Tool for protein sequences) [9]. Although it is the most employed approach in practice, it offers no way of assigning a function to proteins with no homologous sequences.

Recently, ML algorithms for the classification of enzymes that learn patterns and make predictions from protein sequence data gained a lot of attention [3, 8, 10–17]. In many of these studies, the first aim is to convert input samples, specifically amino-acid sequences, into a numerical format, referred to as features. Various approaches have been employed to generate these features, including methods based on homology [3, 8, 11, 12, 18], physicochemical properties [8, 18], presence of functional domains [8, 10], amino-acid sequence properties [12, 18], the raw amino-acid sequence [3, 8, 13], or the usage of learned representations from a protein language model [14, 16].

Over the years, several ML models were devised to learn from these numeric representations. Between 2007 and 2018, the predominant models included *k* Nearest Neighbors (*k*-NN) [10, 18], Random Forests (RF) [12], and Support Vector Machines (SVM) [18]. From 2018 onward, the focus shifted to Deep Learning (DL) methods, best suited for tasks involving large data volumes, aligning with the rapid expansion of databases of curated enzymes [8, 19]. DEEPre [8], one of the earliest DL models, tackles the issue of nonuniform feature dimensionality, which arises when features depend on sequence length, by developing a hybrid DL architecture. This combines Convolutional Neural Networks (CNN) with Long-Short Term Memory (LSTM) networks to learn latent protein representations that depend on sequence length, like one-hot encoding and Position-Specific Scoring Matrix (PSSM), and incorporates a fully connected Deep Neural Network (DNN) to learn length-independent features, such as the presence of Pfam [20] functional domains. Ultimately, the latent representations are concatenated and processed by a DNN to predict the EC number. D-SPACE [21], another approach based on CNNs, was developed as a protein annotator capable of performing various tasks using a single model, including classifying EC numbers, Gene Ontology (GO) terms and protein families. DeepEC [3] was developed later, combining CNNs with similarity-based searches performed with DIAMOND [22]. More recently, ProteInfer [13] was developed, leveraging deep dilated CNNs and data parallelism to train with full-length sequences of amino acids and offering an interpretation of the predictions through class activation mapping.

Although these DL models show great promise by themselves, approaches combining the model predictions and those of other approaches, such as pair-wise alignments with BLASTp [13] and DIAMOND [3, 17], or multiple sequence alignments [14] were explored. The systematic usage of these approaches suggests that DL models provide sub-optimal results that are later compensated by alignments' predictions. The authors of ProteInfer provided an analysis of the results of BLASTp and those of DL models, reaching the general conclusion that BLASTp performs slightly better than ProteInfer deep dilated CNN, and an ensemble of the two delivers performance surpassing that achieved by the individual techniques [13]. On the other hand, CLEAN showcased higher predictive capabilities than BLASTp for independent small datasets of understudied enzymes [16] by mitigating the unbalanced distribution of EC numbers within the training dataset through the application of contrastive learning.

Since Large Language Models (LLMs) have gained significant traction, protein LLMs have been used for EC number prediction [14–17]. These models are based on the transformer network, which employs self-attention mechanisms to weigh the importance of different parts of a sequence, enabling to capture long-range dependencies.

Two types of pre-trained LLMs have been used for EC number prediction so far: Evolutionary Scale Modeling (ESM) [23, 24] and ProtBERT [25]. ESM models are transformer networks pre-trained with UniProtKB data [19] and were used for feature extraction, involving taking the outputs from a layer within these models and using them as features (embeddings) for predicting the EC number [14, 16]. ESM models have been proven to be highly competitive amongst LLMs for protein representation and function prediction [26]. On the other hand, ProtBERT is a transformer network pre-trained with data from UniProtKB and the BFD database [27] and was fine-tuned for EC number prediction [15, 17]. Indeed, although many studies have investigated ESM and ProtBERT models for EC number prediction [14–17], a thorough comparison of these LLMs as feature extractors has not yet been conducted. This lack of comparative analysis leaves unanswered questions about the relative effectiveness of different LLMs in this specific application.

Furthermore, it is important to recognize that the methodologies discussed in these studies typically do not demonstrate their performance independently without the aid of alignments. Moreover, none of the above investigations have thoroughly compared LLMs to BLASTp on large test sets. This leaves open questions about LLMs' performance against BLASTp, their potential advantages, and where LLMs might fall short or require further development to meet or exceed BLASTp's capabilities.

This study presents an in-depth evaluation of advanced protein representations and DL architectures for predicting EC numbers. By addressing the noticeable gap in existing research, we aim to clarify their relative effectiveness. To this end, we have designed a robust experimental framework to evaluate DL models tailored for EC number prediction, trained with embeddings of ESM2, ESM1b, and ProtBERT. Moreover, we provide implementations of DeepEC and D-SPACE for comparison. Since these methods use one-hot encodings as input rather than LLM embeddings, they offer a useful way to compare different input types. Through this comprehensive approach, we aim to critically assess the performance of both BLASTp and DL models, thereby contributing valuable insights into the effectiveness of DL models, enhanced by state-of-the-art LLM embeddings or one-hot encodings, compared to a traditional, alignment-based protein function prediction methodology.

Materials and methods

Defining EC number classification problem

EC number prediction was defined as a multi-label classification problem incorporating promiscuous and multi-functional enzymes (with more than one EC number). Let X be the set of protein sequences and Y be the set of EC numbers. Each protein sequence x_i in X has an associated binary label vector y_i of length $|Y|$, where $|Y|$ is the total number of unique EC numbers. Each vector element y_{ij} is either 0 or 1, indicating whether protein sequence x_i is associated with the EC number j in Y. It is worth noting that all levels were included in the resulting matrix as proposed by [10], so if an enzyme is assigned the EC numbers 1.1.1.1 and 4.1.1.1, ones are placed in y_i at the positions corresponding to 1, 1.1, 1.1.1, 1.1.1.1, 4, 4.1, 4.1.1 and 4.1.1.1. Herein, we take a global approach for hierarchical multi-label classification, where a single classifier is challenged to predict the entire hierarchy of labels and their relationships [28].

UniProtKB data extraction and processing

The SwissProt (manually annotated), TrEMBL (automatically annotated) protein data and UniRef90 clusters were downloaded from UniProtKB [19] in XML format in February 2023. Then, an XML parser was developed to extract the EC numbers, protein sequences, and identifiers of SwissProt and TrEMBL. Only the UniRef90 cluster representatives were kept in the datasets, as first proposed by [10]. By definition, UniRef cluster representatives are chosen based on entry quality, annotation score, organism relevance, and sequence length, enhancing biologically relevant information retrieval. This procedure ensures that enzymes that share more than 90% identity with these representatives are removed from the dataset, thus avoiding redundancy and prioritizing entries with better annotation.

Afterwards, for all EC numbers with less than 100 sequences, we added data from TrEMBL, if available. The pipeline then systematically removes the underrepresented EC numbers, numbers with less than 100 sequences assigned, and provisional EC numbers with the letter "n" in the last level. The final dataset included 380,811 enzymes, categorized under EC numbers as follows: 7 for level 1, 69 for level 2, 230 for level 3, and 2465 for level 4.

Data splitting

Dividing data into training, validation and test sets poses challenges with highly sparse matrices of labels like the ones in this work. It is crucial to ensure that all labels are well-distributed across the sets. To address this, we opted for a split of 60% for training and 20% each for validation and testing, firstly using an iterative approach of stratification of multi-labelled data from the *scikit-multilearn* package v. 0.2.0, described by [29].

Upon running this tool, the stratification was unsuccessful, as a few EC classes had no sequences assigned in the test or validation sets. We, thus, developed an algorithm (Supplementary Algorithm S1) that iteratively transfers a calculated portion of sequences associated with EC numbers from the training set to the test or validation set. The algorithm targets EC numbers where the test-overall data proportion is lower than 15%, a value determined as optimal after some tests. It calculates the number of samples to be moved to compensate for this gap. This ensures that underrepresented EC numbers are more evenly distributed across the datasets, as corroborated in Supplementary Table S1. All the resulting dataset information can be referred to the Supplementary Tables S1 and S2. This process of stratified splitting may lead to certain EC number classes having fewer than 100 sequences each in the resulting datasets.

Complementarily, we developed a train-test split strategy that focuses on identity thresholds, while resampling the data into five folds. We implemented six distinct identity thresholds (90%, 80%, 70%, 60%, 50% and 40%) using CD-HIT [30] to ensure that no sequences in the training set share more than the specified identity threshold with those in the test set. In total, we got 30 train-test combinations (six identity thresholds times the five folds). This approach allowed us to evaluate how similarity affects the performance of each method.

Designing the evaluation datasets

In this work, we evaluated the predictive capacity of the models on five different datasets. The first three are progressively refined datasets derived from the initial test set based on UniProtKB data and obtained from the employed data-splitting strategy. The first dataset included the entire test set, while the second was refined to proteins with an EC number corroborated by at least one literature reference, termed the evidence-level dataset. While this test set may contain some automatically inferred annotations, we believe that its reliance on manually curated SwissProt entries with literature citations strikes a good balance between annotation quality and dataset size, especially since manually annotating all entries would be too time-consuming, more information is included in the Supplementary Information. The third dataset, taken from the evidence-level dataset, focused on promiscuous and multi-functional enzymes. Two additional, fully independent datasets were considered for evaluation: one from [31] with 149 bacterial enzymes, known for its experimentally validated yet challenging data, and another comprising 36 incompletely annotated halogenases identified from the UniProtKB database, further curated by [16]. These two datasets have been used for performance comparisons [16, 17]. Supplementary Table S2 contains the number of enzymes and EC numbers per dataset.

Language models embeddings for enzyme sequences

In this study, we used three different pre-trained LLMs as feature extractors of the enzyme amino acid sequences. By extracting the outputs of the last layer of these models and applying an element-wise mean over it, we obtained embeddings for the EC number prediction. The models used as feature extractors were ProtBERT [25], ESM1b [23], and ESM2 [24], trained on protein sequences without label supervision. For details on the pre-processing of protein sequences, refer to Supplementary Information, Figure S1 and Table S3.

Let us denote the output of the transformer's last layer as $\mathbf{M}$, a matrix of size $t \times n$, where t represents the number of amino acids and n the dimensionality of the feature representation for each token. In protein LLMs, each token represents an individual amino acid, serving as the smallest data unit processed by the model. Moreover, a special "CLS" token is always added at the beginning of the sequences, and it does not represent any amino acid. For this reason, we exclude it from our calculation. Therefore, we consider the submatrix $\mathbf{M}'$ of $\mathbf{M}$, excluding the first row, represented as:

$$\mathbf{M}' = \begin{bmatrix} m_{1,0} & m_{1,1} & \cdots & m_{1,n} \\ m_{2,0} & m_{2,1} & \cdots & m_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ m_{t,0} & m_{t,1} & \cdots & m_{t,n} \end{bmatrix} \tag{1}$$

where, $\mathbf{M}_{i,j}$ represents the feature value for the j-th dimension of the i-th token. The element-wise average for each dimension across all tokens is then computed as follows:

$$E_j = \frac{1}{(t-1)} \sum_{i=1}^{t} m_{i,j} \quad \text{for } j = 0, 1, ..., n$$

where E_j is the average of the j-th dimension across all tokens. The resulting vector of these averages, E, which is of dimension n, represents the element-wise average of the last layer's output:

$$E = [E_0, E_1, ..., E_n]$$

E provides a single representation that captures the mean features across all tokens in the last layer of the transformer. Seven distinct embeddings were calculated, employing ProtBERT, ESM1b, and five variations of ESM2. The resulting dimensions for the used LLM embeddings are detailed in Supplementary Table S4.

Model architectures and hyperparameters

First, we trained baseline models with each of the embeddings to establish a foundation for evaluating the effectiveness and impact of more complex models. The baseline linear models were shallow networks with only input and output layers. The former has as many neurons as the dimension of the feature vector, and the latter as many neurons as labels (EC numbers to predict) with a sigmoid activation function. The loss function used was binary cross-entropy.

Then, we added hidden layers with the Rectified Linear Unit (ReLU) activation function, followed by batch normalization, to the baseline models to assess the performance compared with the baselines. In this context, we performed an architecture search of four trials for the seven embeddings. Architectures 1 and 2 featured a single hidden layer with 640 and 2560 neurons, respectively, while architectures 3 and 4 had two hidden layers, with neuron counts of 640 and 320 for the former and 2560 and 5120 for the latter. We trained these networks with the Adam optimizer with a learning rate of 0.001, as after a few preliminary experiments on the training set, we found this value to be the most suitable value for the learning rate. The training process was conducted for 30 epochs. For terminology, these models will be referred to as "DNN", followed by the name of the language model that provided the input embedding. For clarity, models utilizing LLM embeddings will henceforth be designated as DNN-LLM models, differentiating them from models that do not employ such embeddings.

Among all the trained models, we chose those with the highest performance on the validation set based on their embeddings and architecture. For example, if architecture 4 delivered the best performance with ESM1b among all tested architectures for DNN ESM1b, we would select it. As ESM2 has five different embedding variations, if the best architecture of DNN ESM2 3B outperformed DNN ESM2 650 M, we proceeded with the best architecture of DNN ESM2 3B. These top-performing models were then retrained using training and validation data. Complementarily, we created a model ensemble, a voting classifier consisting of ESM2 3B, ESM1b, and ProtBERT best models. The consensus class prediction for this ensemble is based on hard voting, i.e. the final prediction is determined by a simple majority vote. For terminology, the ensemble of top-performing models is referred to as 'Models ensemble'.

Then, we reimplemented the D-SPACE and DeepEC approaches, as they encode enzymes with one-hot encoding rather than LLM embeddings. Comparing the performance of these models with ours will provide valuable insights into the relative effectiveness of LLM embeddings as encodings of enzyme sequences for EC number prediction. D-SPACE [21] comprises three repeated CNN components and a fully connected network to generate a prediction vector. We reimplemented the EC numbers prediction part, which consists of three CNNs with two layers of 1D convolutions, followed by batch normalization in each, a max pooling layer at the end, and three fully connected layers and an output layer with as many neurons as the number of labels to predict.

In DeepEC [3], the training process involves three CNNs: CNN-1, which assesses whether a protein is an enzyme; CNN-2, which predicts EC numbers up to the third level; and CNN-3, which extends the prediction to the fourth level. The final EC number prediction is derived when the outputs of CNN-2 and CNN-3 are consistent. However, if either CNN-2 or CNN-3 fail to provide a reliable prediction, DIAMOND [22] is employed to make the EC prediction. In our revised implementation of DeepEC, the architecture exclusively incorporates the CNN-3 model.

A threshold of 0.5 was used for the output layer to determine positive predictions across all models. By setting the threshold at 0.5, any output probability equal to or greater than this value is classified as positive, while probabilities below it are classified as negative. This approach ensures consistency in evaluation across all models.

All models were implemented and trained using the *pytorch* package v. 2.1.2. Supplementary Section 4 presents more details on the implementation.

Pair-wise alignments

BLAST is designed to compare a query sequence against a database of reference sequences to find regions of similarity. BLAST uses the E-value (Expectation value) as a measure of the significance of a sequence alignment. The E-value is the expected number of random hits with a similar or better score that one would encounter in a database search for the given query sequence purely by chance.

An experiment was conducted using BLASTp to annotate the test datasets and compare the performance with the DL models. We merged the train and validation sets to serve as the reference database for BLASTp. The hit with the lowest E-value was selected, and the EC number was assigned directly from the corresponding sequence. Although we used a fully automated procedure, we acknowledge that results from both BLAST and DL models could be improved with manual annotation. However, this is beyond the scope of this work and would be laborious on a large scale. The BLAST E-value cutoffs were set to 10^{-5}, as suggested by [32] as a typically used value. NCBI BLAST version 2.12.0+ was used. A full list of parameters can be found in Supplementary Table S5. We further created a combined ensemble of the top-performing models and BLASTp, based on hard voting. The system chooses the positive prediction when there is a tie of predictions. In terms of terminology, the ensemble that includes both the top-performing models and BLASTp is called 'Models + BLASTp.'

Metrics

We employed several metrics to evaluate the performance of the models. These metrics provide a comprehensive understanding of the model's predictive capabilities, especially when dealing with multi-labelled datasets. The key metrics are the F1 score macro (mF1), F1 score weighted (wF1), Recall macro (mRecall), Recall weighted (wRecall), Precision macro (mPrecision), Precision weighted (wPrecision) and accuracy.

The mF1 is a metric that calculates each label's harmonic mean of precision and recall and then averages them to obtain a single score for the entire dataset. The same principle is applied to mRecall and mPrecision, where the average of precisions and recalls across labels is computed. Another way of calculating these metrics is using the weighted average version, where the scores of each label are multiplied by the relative frequency of that label in the dataset. These metrics were calculated using the *scikit-learn* package version 1.2.0. Their definitions and formulas can be found in the Supplementary Table S6.

Finally, we evaluated the consistency of the hierarchical predictions of our global multi-label classifier. As a hierarchical system, the EC numbers must be predicted as the following formulation: if a model predicts a more specific label (like 1.1.1.1), it also must predict all the corresponding higher-level labels in the hierarchy (like 1.1.1, 1.1, and 1 in this case). If the prediction aligns with the stated formulation, it is considered consistent; otherwise, it is considered inconsistent. Accordingly, we formulated a metric called Hierarchical Consistency Error (HCE), defined as follows:

$$HCE = \frac{\sum_{i=1}^{T} I(s_i)}{T} \tag{2}$$

where T is the number of enzymes in the dataset, and $I(s_i)$ is an inconsistency indicator function for each enzyme s_i, returning one if there is at least one inconsistency in the hierarchical predictions for enzyme s_i, and 0 otherwise. Let $P(s_i)$ be the set of labels predicted for enzyme s_i. For each label l in $P(s_i)$, let $H(l)$ be the set of hierarchical higher-level labels of l. Then, the inconsistency indicator function $I(s_i)$ for enzyme s_i is:

$$I(s_i) = \begin{cases} 1, & \text{if } \exists l \in P(s_i) \text{ such that } H(l) \not\subseteq P(s_i) \\ 0, & \text{otherwise} \end{cases} \tag{3}$$

As EC numbers can be represented as a hierarchical ontology in a directed acyclic graph (DAG), where each node represents an EC, we applied metrics specific to biological hierarchical ontologies. In this definition, level 1 nodes branch into more specific subclasses at level 2, further specified by sub-subclasses at level 3, and finally detailed at a deeper level (level 4), with each level's definition dependent on its predecessor, establishing a cascading specificity from general to precise.

Accordingly, metrics for evaluating how confident each prediction is within an ontology were considered. To interpret these predictions accurately, it is essential to consider the *remaining uncertainty*, *misinformation*, and *S-min* (semantic distance) in relation to a decision threshold. Predictions above or equal to the decision threshold are classified as positive, while those below are negative. These metrics were first described in [33] and implemented in Python in CAFA-evaluator [34]. We adapted this implementation for the EC system.

Remaining uncertainty reflects the information missing from a protein's true annotation that is not captured by the prediction DAG. Formally, the remaining uncertainty is the total information content of nodes in the ontology that exists in the ground truth DAG but is absent in the prediction DAG.

Misinformation reflects the total information content of nodes along incorrect paths in the prediction DAG, quantifying how misleading the predicted annotation is.

Semantic distance or *S-min* helps by combining these two measures and is defined as the minimum distance from the origin to the "remaining uncertainty vs. misinformation" curve. It does not just look at how much information is missing or incorrect individually but balances both to give a single performance score. When semantic distance is low, it indicates that the prediction is both complete (not missing key information) and accurate (not misleading with incorrect information). This helps rank algorithms in terms of both their accuracy and completeness.

Statistical methods

We utilized the Wilcoxon Signed-Rank (WSR) test to assess the statistical significance of the differences in metric values between the methods across the test sets. In this scenario, the samples comprise the metric values generated by two methods for each EC number within the multi-label classification framework. The goal is to assess the significance of any differences in the median metric values across all EC numbers. In applying the WSR test to evaluate differences in metric values across various identity thresholds and EC numbers, we also incorporated a False Discovery Rate (FDR) correction for multiple hypothesis testing. We applied the Benjamini-Hochberg (BH) FDR correction to the p-values. We deemed a corrected p-value of less than 0.001 sufficient for rejecting the null hypothesis. For more information on how we applied the statistical methods, please refer to Supplementary Information.

Results

DNN-LLM perform generally on par with BLASTp

Our workflow included a benchmark of models trained with ESM and ProtBERT embeddings compared to BLASTp. Initially, linear baseline models were established, setting a reference for evaluating more complex models. These baselines were then enhanced

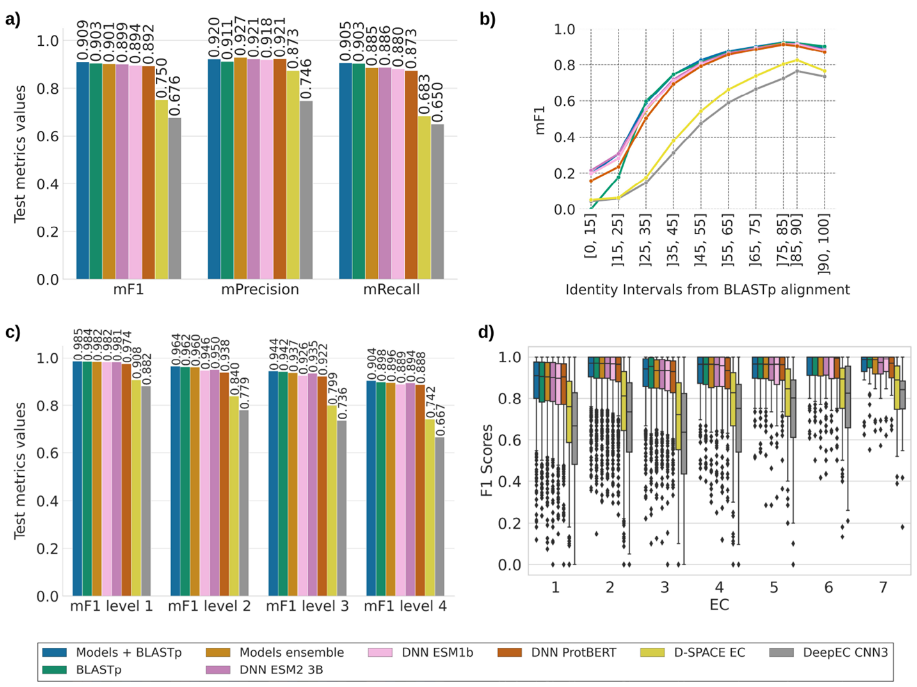

Fig. 1 Results and evaluation using the whole test set. **a** mF1, mRecall and mPrecision of all models. **b** mF1 score for different identity intervals. **c** mF1 for the different levels of EC numbers (i.e. level 1: X.; level 2: X.X.; level 3: X.X.X.; level 4: X.X.X.X). **d** Distribution of F1 scores across level 4 EC classifications (X.X.X.X format) for each method. This boxplot displays F1 score distributions organized by EC category of level 1 (1–7), capturing the performance of each method across specific enzyme functions within these categories (e.g. EC 1 includes F1 scores for subcategories (1.X.X.X), EC 2 includes F1 scores for subcategories (2.X.X.X), and so forth

with hidden layers and batch normalization and compared against reimplementations of architectures like D-SPACE and DeepEC. A clear enhancement in performance was verified from the baseline models to their counterparts, as shown in Supplementary Figure S2.

Figure 1a demonstrates that models using LLM embeddings substantially outperformed those that did not, indicating their advantage for the EC prediction task. This was evident compared to DeepEC CNN3 and D-SPACE EC models, which relied on a one-hot encoding of amino acid sequences. Comparing the performance of BLASTp and DNN-LLM models reveals that BLASTp was slightly better than the DNN-LLM models in terms of mF1 score, with a difference of 0.011 from the worst DNN-LLM model. Figure 1c reveals a consistent trend across all EC levels.

Regarding mPrecision and mRecall, the results indicate that all models, except DeepEC CNN3 and D-SPACE EC, surpass BLASTp in mPrecision, while BLASTp demonstrates better mRecall values. This observation is consistent with the Recall-Precision curves reported by [13]. They suggest that BLASTp's higher recall may be due to its direct utilization of the entire training set, in contrast to neural networks that condense this data into a limited set of weights [13]. On the other hand, BLASTp's comparatively lower precision could result from evolutionary changes in sequences, where proteins might closely match parts of a training sequence but lack key regions crucial for the enzyme function [13]. Additionally, we can observe that combining DNN-LLM models with

BLASTp in an ensemble approach (Models + BLASTp) leads to a better performance than using either approach independently (Fig. 1c), a pattern that is consistent across EC levels, confirming the findings from [13].

Figure 1b shows the mF1 scores of the models applied to subsets of the test set organized by sequence identity levels with the training set, calculated by BLASTp. The findings indicate that within the lowest identity interval (below 15%), where BLASTp cannot make any predictions, even the least accurate DL model, DeepEC CNN3, still achieved a mF1 score of 0.04, whereas the best model, DNN ESM2 3B, reached 0.21. Below 25% of identity, the DNN-LLM models provided better predictions than BLASTp; however, they performed marginally worse than BLASTp for higher identity levels.

When no similar sequences were available, BLASTp performed worse than DL models. This is increasingly relevant in functional annotation pipelines, as more sequences from previously uncharacterized organisms become available. For example, current alignment-based tools are unable to determine the functions of one-third of microbial protein sequences [35].

Figure 1d provides a detailed view of F1 scores across level 4 EC classifications (X.X.X.X format), where results are aggregated by EC category of level 1 (1–7). In this boxplot, we can have a better perspective on the variability of performances of each method per EC category. The standout finding is clear: DNN-LLM models and BLASTp consistently outperform CNN-based models, and the performance difference is statistically significant (p-value < 0.001 in the WSR test). At the highest precision level (level 4), BLASTp takes a narrow lead, surpassing only EC 3 (hydrolases). Generally, all DL models fell shorter in predicting ECs across EC 3 than BLASTp. Furthermore, BLASTp performs significantly better than other models like DNN ESM1b and DNN ProtBERT for EC 1, 2, and 3.

When comparing our top combination, Models + BLASTp, with BLASTp alone, the combined approach shows a clear advantage, particularly for EC 2 (transferases). Models + BLASTp also outperformed individual DNN-LLM models across the board, except for EC 7 (translocases), EC 6 (for DNN ProtBERT and DNN ESM1b), and EC 5 (for DNN ProtBERT).

Interestingly, ECs 6 and 7 emerged as the 'easier' categories, with median F1 scores at or near one across all levels and impressively low variability, indicating consistent predictions. EC 2 is another level which was predicted well by most predictors, with low variability of median F1 scores. EC 1, however, presents the greatest challenge, with the lowest median F1 scores and 75th percentiles across the board and all levels.

The models were then evaluated on the subsets of enzymes with additional information, such as literature references (evidence-level dataset), enzyme promiscuity and multi-functionality (see Methods). We identified the wF1 score as the most suitable metric for these cases, given their smaller size and the evident class imbalance, further detailed in Supplementary Figures S3 and S4. Tables 1 and 2, and Supplementary Figure S5 present the results for these datasets. The results reveal that BLASTp generally outperforms individual DNN-LLM models and the Models ensemble at EC levels 2, 3, and 4, albeit the margin of performance difference is not substantial. In almost all cases, except for level 4 in the evidence-level dataset, the Models + BLASTp ensemble performed better than any of the methods used independently, following the trends

Table 1 wF1 scores for the evidence-level dataset. The levels refer to the different hierarchical levels of the EC number system (i.e. level 1: X.; level 2: X.X.; level 3: X.X.X.; level 4: X.X.X.X). The highest-performing result at each level is indicated in bold

	wF1			
Models	**lvl 1**	**lvl 2**	**lvl 3**	**lvl 4**
DNN ESM2 3B	0.967	0.945	0.928	0.862
DNN ESM1b	0.969	0.943	0.922	0.852
DNN ProtBERT	0.953	0.922	0.902	0.842
DeepEC CNN3	0.818	0.719	0.679	0.587
D-SPACE EC	0.854	0.780	0.751	0.651
BLASTp	0.967	0.948	0.931	**0.878**
Models ensemble	0.970	0.946	0.928	0.861
Models + BLASTp	**0.973**	**0.952**	**0.936**	0.876

Table 2 wF1 scores for the promiscuous and multi-functional enzymes dataset. The levels refer to the different hierarchical levels of the EC number system (i.e. level 1: X.; level 2: X.X.; level 3: X.X.X.; level 4: X.X.X.X). The highest-performing result at each level is indicated in bold

	wF1			
Models	**lvl 1**	**lvl 2**	**lvl 3**	**lvl 4**
DNN ESM2 3B	0.925	0.910	0.880	0.742
DNN ESM1b	0.941	0.907	0.880	0.735
DNN ProtBERT	0.932	0.888	0.856	0.742
BLASTp	0.940	0.913	0.893	0.770
DeepEC CNN3	0.781	0.685	0.626	0.447
D-SPACE EC	0.839	0.753	0.699	0.502
Models ensemble	0.937	0.912	0.883	0.748
Models + BLASTp ensemble	**0.949**	**0.923**	**0.898**	**0.779**

observed in the overall test set. Accordingly, among the DNN-LLM models, DNN ESM2 3B emerged as the top performer overall, except for level 1, where DNN ESM1b outperformed it.

While these trends remained consistent, there was a noticeable decline in performance for the dataset comprising promiscuous and multi-functional enzymes (those with multiple EC numbers). This indicates that making predictions for enzymes with more than one EC number is more challenging, highlighting the complexities associated with enzyme promiscuity and multifunctionality.

BLASTp outperforms DNN-LLMs across identity thresholds

Next, we set out to investigate whether the identity thresholds based on which the enzymes are split into train and test datasets affect the observed differences in models' performances. To this end, we performed a 5-fold cross-validation with train-test splits based on different identity thresholds. Figure 2 compares the mF1 score of various models across different identity thresholds, demonstrating how each model's performance improves as the identity threshold increases from 40% to 90%.

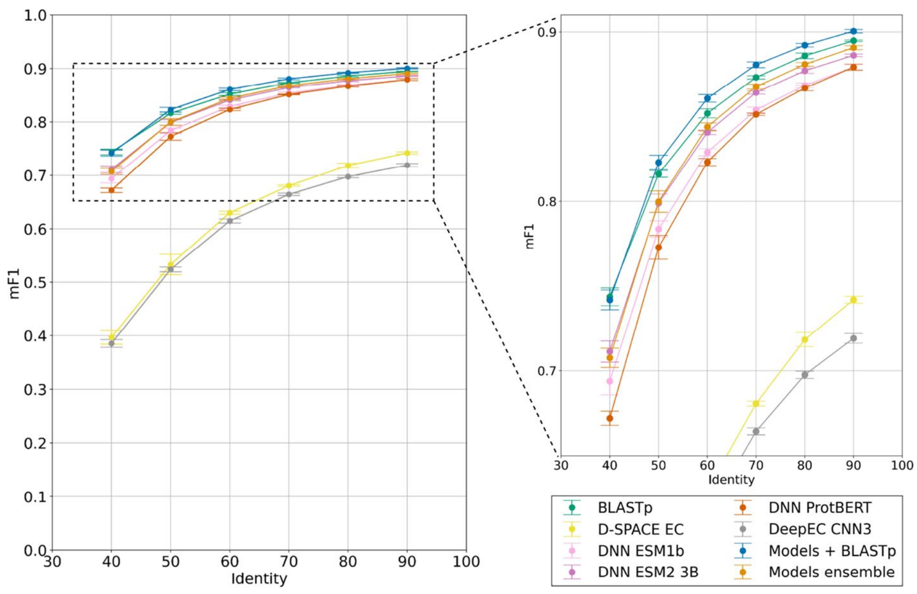

Fig. 2 The mF1 scores are presented as a function of various identity thresholds in a train-test split setting across five folds. On the left, a comprehensive overview of the results is displayed on a scale from 0 to 1, showing the mF1 scores. On the right, a zoomed-in version of the plot on the right provides a more detailed view of these results

BLASTp (green line) consistently shows strong performance across all identity levels, achieving significantly higher (p-value < 0.001 in a WSR test) mF1 scores than DL models at nearly all identity levels, lagging behind only Models + BLASTp.

The DNN-LLM models generally show competitive performance, especially as identity increases, but tend to fall slightly below BLASTp. They perform reasonably well at mid-identity levels, with performance stabilizing as identity reaches 90%. Among these, DNN ESM2 3B is clearly the best at all identity thresholds, significantly outperforming the rest (p-value < 0.001 in a WSR test), being followed by DNN ESM1b which significantly outperforms DNN ProtBERT except for the identity threshold of 90%.

D-SPACE EC and DeepEC CNN3 models (yellow and grey lines) show significantly lower performance than the other methods (p-value < 0.001 in a WSR test), particularly at lower identity levels. Their mF1 scores improve with increasing identity but remain below those of BLASTp and the DNN-LLM models.

The combination of models with BLASTp (Models + BLASTp) achieves the highest overall performance. This method significantly surpasses all other approaches at every identity threshold (p-value < 0.001 in a WSR test), except for the 40% threshold, where it is slightly outperformed by BLASTp alone, but with no significant differences.

DNN-LLMs successfully learn the hierarchy of the EC system

Figure 3 shows how each evaluated model performs in terms of learning the hierarchical structure of the EC system at different thresholds of positive classification (Fig. 3a and c) and using the threshold of 0.5 (Fig. 3b).

Figure 3a compares models based on remaining uncertainty (missing true information) and misinformation (incorrect predicted hierarchy paths) at different positive classification thresholds, with semantic distance (S-min) combining these measures. DNN

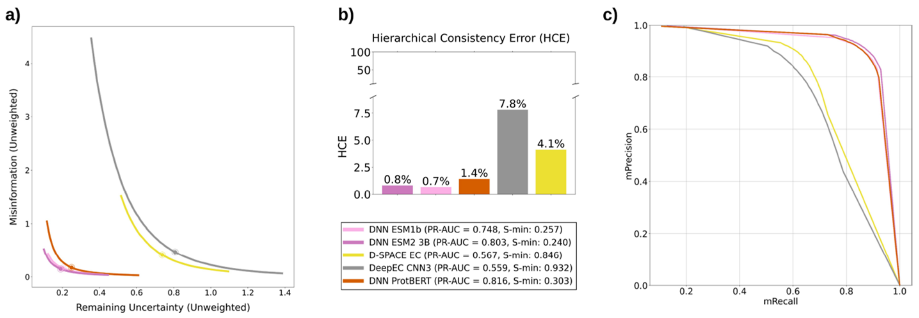

Fig. 3 The performance of each evaluated model in learning the hierarchical structure of the EC system using the whole test set., with panels **a** and **(c)** showing results across different positive classification thresholds, and panel **b** focusing specifically on the threshold of 0.5. **(a)** Remaining uncertainty and misinformation at different thresholds. S-min is represented in the plot as a circle. **(b)** HCE for all DL models. **(c)** mRecall and mPrecision across different positive classification thresholds. The area under the curve is shown in the legend

ProtBERT, DNN ESM2 3B, and DNN ESM1b (DNN-LLM) have the lowest S-min values, indicating that they capture the most complete and accurate information, minimizing both missing and misleading annotations. In contrast, DeepEC CNN3 and D-SPACE EC show higher S-min scores, suggesting that they miss more true information or include more incorrect paths, making them less effective overall. Among DNN-LLMs, DNN ESM2 3B stands out as the most optimal model with an S-min of 0.240 and with the most left-cornered remaining uncertainty-misinformation curve, outperforming DNN ESM1b and DNN ProtBERT.

Besides the metrics described above, we recognize that in hierarchical multi-label classification, understanding the relationships between labels is also key to learn the hierarchy. Figure 3b shows the Hierarchical Consistency Error (HCE) for each DL model at a positive classification threshold of 0.5, indicating that DL models utilizing LLM embeddings, particularly ESM-like, substantially outperformed others in terms of HCE. This shows that LLM-based models can achieve higher mF1 scores and be more proficient at learning the EC hierarchy, with ESM-like embeddings outperforming ProtBERT. However, applying simple heuristics to these predictions could eventually reduce this error and improve the results. For instance, if one model predicted the EC number as 1.1.1.1, one could apply a heuristic to assume that the precedent levels are 1, 1.1 and 1.1.1.

Since a prediction threshold of 0.5 (the output of the sigmoid function in the output layer) may not fully capture how the models perform in terms of mPrecision and mRecall and learn the hierarchy at different levels, we examined their behaviour across a range of prediction thresholds. Figure 3a shows precision-recall (PR) curves across these thresholds. The PR curve and respective Area Under the Curve (AUC) of DNN-LLM models show that these models outperform CNN-based models (D-SPACE and DeepEC). Generally, the DNN-LLM-based models show strong initial mPrecision values from 0.96 to 0.91 when mRecall is below 0.85, but slowly decrease to a minimum of 0.85 as recall approaches 0.90, with slight differences in DNN ESM2 3B, where the decrease is less steep.

Moreover, a closer look at the results shows that DNN ProtBERT and ESM2 3B both achieve very high mPrecision and mRecall values, with DNN ESM2 3B slightly outperforming DNN ProtBERT in both metrics at multiple thresholds. However, PR-AUC is not solely a function of the highest precision or recall values. It captures the trade-off between mPrecision and mRecall across all threshold levels. The differences in the progression of mPrecision and mRecall rates across thresholds result in a PR curve for DNN ProtBERT that covers slightly more AUC in certain conditions. The major difference is when the values of mRecall are lower than 0.8, DNN ProtBERT maintains a higher mPrecision than DNN ESM 3B. However, for mRecall above 0.8, DNN ESM 3B mPrecision is higher than for DNN ProtBERT. This suggests that DNN ESM 3B is better suited for higher threshold settings and is likely less sensitive to threshold changes, particularly when mRecall exceeds 0.8. The higher precision DNN ESM 3B achieves at high recall levels indicates that it is more stable in maintaining mPrecision as mRecall increases, which is beneficial when high-recall, high-precision predictions happen, which is the case (see Fig. 1). In contrast, DNN ProtBERT's strength lies in maintaining higher mPrecision at lower recall levels (under 0.8). This means that ProtBERT is more responsive in prioritizing correct predictions when aiming for moderate recall but may lose mPrecision as mRecall pushes closer to 1.0. This characteristic implies that DNN ProtBERT could be advantageous in cases where more selective, lower-recall thresholds are acceptable, but it may not be as stable across higher recall ranges compared to ESM 3B.

DNN-LLMs and BLASTp are complementary

To test whether different methods performed better for different enzyme types, we next compared the performance of BLASTp and DNN-LLM models (in terms of the difference in F1 scores from both methods) for each EC class, spanning all levels ($\Delta F1_i = F1_{\text{DNN-LLM},i} - F1_{\text{BLASTp},i}$, where i is one class of EC numbers), as shown in Fig. 4. This figure indicates that while some enzyme classes have the same F1 scores for both methods, others vary. Figure 4 includes scatter plots illustrating how the differences in F1 scores between the DNN-LLM models and BLASTp relate to the number of sequences in each EC class. Also, bar plots show how many EC classes have F1 score differences surpassing a set threshold.

The analysis indicates that BLASTp demonstrated superior performance for more EC numbers than DNN-LLM models. However, it is important to highlight that each DNN-LLM model evaluated in this study outperformed BLASTp on hundreds of classes. Among these, DNN ESM2 3B emerged as the top performer, annotating 792 classes better than BLASTp, followed by DNN ESM1b and DNN ProtBERT, with 746 and 710 classes. Regarding F1 score differences, the leading model, DNN ESM2 3B, exhibited a minimum increase of 0.1, 0.2, and 0.3 in F1 scores for 107, 14, and 3 classes, respectively. A notable mention is DNN ESM1b, which achieved a significant F1 score improvement compared to BLASTp of 0.4 in two labels (1.1.1.274 and 4.3.2.3). Detailed comparisons of specific EC numbers where DNN-LLM models outperformed BLASTp can be found in Supplementary Table S7.

Figure 4 highlights underrepresented classes, specifically those with fewer than 63 sequences per class (red squares). There are 69 such classes. The plot delineates which classes are more accurately annotated by BLASTp (indicated by values below 0) versus those better annotated by the DNN-LLM models (indicated by values above 0). The

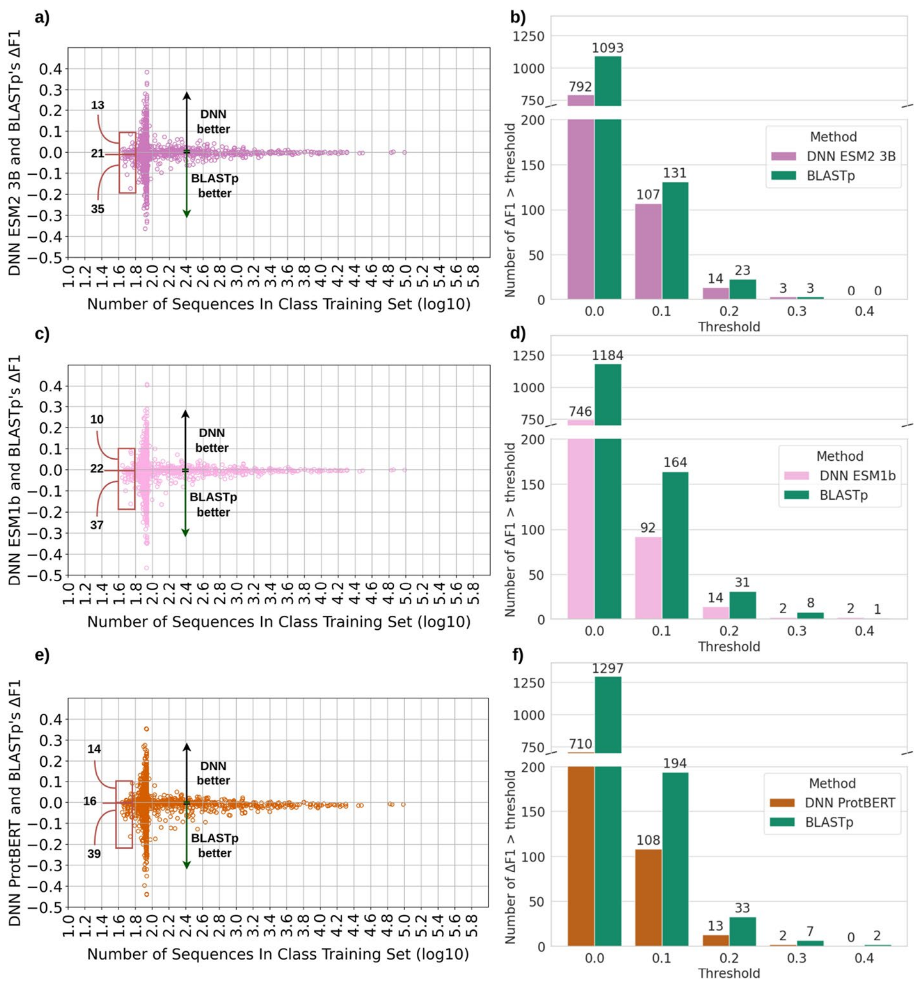

Fig. 4 Comparison of DNN-LLM models' and BLAST's F1 scores. **a,c,e** Distribution of DNN-LLM models' and BLASTp's F1 score differences per number of sequences in the training set ($\log_{10}$), where each dot represents an EC class. **b,d,f** Number of classes with an $\Delta F1_i$ above a given threshold

findings reveal that, among the differentially annotated classes i.e. $\Delta F1_i \neq 0$, the proportion of classes better annotated by DL models is lower for these underrepresented classes (24.9%) than the entire set of classes (38.7%). This suggests that while BLASTp generally annotates more classes effectively, this is more pronounced for classes represented less well in the training set.

DNN-LLMs outperform BLASTp on difficult tasks

To challenge the EC prediction methods with more difficult tasks, we applied them to two published datasets: newly annotated bacterial enzymes [31] and halogenases [16]. Since these datasets are independent and contain enzymes not found in the training sets of any of the following state-of-the-art approaches, we compared our methods with two leading approaches: ProteInfer, a CNN-based method, and CLEAN, a contrastive learning model built using ESM1b embeddings.

Table 3 Model wRecall, wPrecision and wF1 scores for the [31] dataset. The highest-performing result for each metric is indicated in bold

Method	wF1	wRecall	wPrecision
CLEAN	**0.495**	**0.584**	0.467
ProteInfer	0.166	0.138	0.243
DNN ESM2 3B	0.433	0.408	0.535
DNN ESM1b	0.421	0.382	0.587
DNN ProtBERT	0.357	0.316	0.526
Models Ensemble	0.392	0.362	0.525
BLASTp	0.372	0.329	0.499
Models + BLASTp	0.444	0.408	**0.600**

The former comprises enzymes whose functions were recently determined through a high-throughput experimental genetic study [31]. The authors provided new EC numbers for enzymes where prior annotations in the SEED or Kyoto Encyclopedia of Genes and Genomes (KEGG) databases were either incorrect or inconsistent. Given this context, this dataset is likely enriched with proteins whose functions pose a significant challenge for computational assessment, making it invaluable for testing and refining computational annotation methods. For evaluation purposes, wRecall, wPrecision, and wF1 scores at level 4 were exclusively employed to address the significant class imbalance and to align with the metrics used by [16], facilitating comparisons with other tools.

BLASTp performs worse than all methods except DNN ProtBERT (Table 3). Moreover, ProteInfer performs worse than all methods, which reinforces the idea that LLM-based models outperform CNN-based models. The custom models developed for this work, DNN ESM2 3B and DNN ESM1b, exhibit strong performance. The Models ensemble shows balanced performance across all metrics but does not outperform the individual DNN ESM2 3B model. The Models + BLASTp ensemble outperforms other methods in wPrecision, including CLEAN. On the other hand, CLEAN delivers better wF1 and wRecall than any other approach. This shows that models trained with LLM embeddings are promising to (re-)annotate enzymes that are difficult to predict or have inconsistent annotations across biochemical databases.

The second challenging dataset comprises 36 halogenases, a group of incompletely annotated enzymes identified from the UniProtKB database. The dataset presents a significant challenge since the halogenase family is relatively understudied, and the existing annotations in databases are often incomplete or conflicting, complicating the task of accurate classification. Despite these challenges, expert curation was conducted by [16], leading to the confident assignment of specific EC numbers to all 36 halogenases in the dataset. To assess method performance, accuracy across all levels was computed, given the lesser degree of imbalance, ensuring alignment with the evaluation criteria outlined in [16] and enabling comparative analysis with alternative tools. Furthermore, given the small sample size in this dataset, binomial tests were conducted to compare the rates of correctly and incorrectly annotated proteins for each EC number at each level. The expected proportions of correctly annotated proteins were estimated based on the overall number of correctly annotated proteins within each level.

Table 4 Model accuracies for the halogenases dataset. The highest-performing result for each level is indicated in bold

Method	Accuracy			
	level 1	level 2	level 3	level 4
CLEAN	**1.000**	**0.972**	**0.944**	**0.944**
ProteInfer	0.611	0.388	0.083	0.694
DNN ESM2 3B	0.917	0.917	0.500	0.805
DNN ESM1b	0.917	0.917	0.500	0.833
DNN ProtBERT	0.805	0.917	0.528	0.750
Models Ensemble	0.917	0.917	0.500	0.833
BLASTp	0.750	0.639	0.194	0.583
Models + BLASTp	0.917	0.917	0.555	0.833

[a] This assessment was performed based on the ground truth derived only from the manual curation in [16]

A halogenase refers to an enzyme that incorporates halogen atoms (such as chlorine, fluorine, or bromine) into organic compounds, often for catalyzing halogenation reactions that add bioactive properties to compounds [36]. In the context of EC terminology, these enzymes fall under specific EC classifications based on their reaction types, such as haloperoxidases (classified as oxidoreductases, EC 1.11.1.10) or flavin-dependent (1.14.19.x, depending on the substrate), α-ketoglutarate-dependent (1.14.20.x and 1.14.11.x), and S-adenosyl-L-methionine (SAM)-dependent halogenases (which can be classified as transferases, EC 2.5.1.x, depending on their exact substrate, or hydrolases, EC 3.13.1.8) [16].

BLASTp exhibited lower accuracy than most methods, as presented in Table 4. As BLASTp was not able to find homologous sequences for four query sequences from this dataset, this might have affected the results compared to DL models, since DNN-LLMs could generate correct predictions for these sequences. At the same time, DNN ESM2 3B, DNN ESM1b, and DNN ProtBERT varied in their highest accuracies across levels 1 and 2, but all showed decreased performance at levels 3 and 4. The Models ensemble matched the top performances of DNN ESM2 3B and DNN ESM1b at the first two levels but also dropped in accuracy at the lower levels. Models + BLASTp achieved the highest overall across all levels without considering CLEAN's performance. CLEAN obtained the best accuracies across the board, and ProteInfer fell short at all levels compared to all the other methods. Binomial tests revealed that, for DNN ESM2 3B, the confidence interval for the predictions of haloperoxidases (EC 1.11.1.10) was slightly above the expected value of baseline performance at level 4 (see Supplementary Material). Similarly, DNN ESM1b and DNN ProtBERT demonstrated confidence intervals exceeding the baseline expectation for the prediction of α -ketoglutarate-dependent halogenases (specifically those within EC 1.14.11.x). These results suggest that these models exhibit superior annotation performance for these enzyme classes compared to other enzymes at the same level.

In this case, the challenges across different EC levels arise from the need to distinguish broad functional categories (level 1), reaction-related components like solvents, cofactors, and electron acceptors (levels 2 and 3, especially in oxidoreductases), and specific substrates (level 4). CLEAN effectively identifies subtle patterns in the

ESM1b embeddings that indicate substrate-specific information, which helps explain its strong performance at the detailed, substrate-specific level 4 EC classification.

Discussion

This study compared DL models trained with embeddings derived from LLMs and the gold standard method BLASTp. This study delivers a fully reproducible end-to-end pipeline for benchmarking models and protein representations for EC number prediction. We show that using LLMs as straightforward feature extractors leads to better results than models relying on the one-hot encoding of sequences (e.g. D-SPACE, DeepEC and ProteInfer). Generally, DNN-LLM models were marginally outperformed by BLASTp for large datasets with homologous sequences available from databases. While BLASTp demonstrated a slight performance improvement over DL models, it is important to note that it has an inherent advantage in this context. Since it is one of the most widely used annotation methods, it is used to annotate enzymes in UniProtKB, which we use for testing the methods, thus favouring BLASTp. Considering that, DL models trained with LLMs achieve impressive results, with a minimal difference of 0.004 in the mF1 score compared to the one from BLASTp. Despite this slight performance gap, in scenarios where the training set lacked homologous sequences, LLM models outperformed BLASTp, indicating their potential utility in more challenging annotation tasks. Particularly, when BLASTp fails to find an enzyme with more than 25% identity in the training set, its performance declines, and DL models outperform it. Hence, we claim that DL models should be employed below this mark, instead of BLASTp. Additionally, combining BLASTp with DL models can enhance results across nearly all identity intervals. We also confirm that DL models are slightly more precise and have lower recall than BLASTp, as pointed out by [13].

As future work on integrating BLASTp and DL models, a more nuanced approach could be adopted rather than simply favouring DL models over BLASTp based solely on sequence identity. Specifically, an alternative method could be implemented to adjust predictions according to the number, query coverage, sequence identity and other metrics of identified BLASTp hits. This approach could involve using a linear regression model or a similar ML technique to optimize the integration of predictions from both BLASTp and DL models. By leveraging the strengths of each method under different conditions, this strategy has the potential to enhance overall predictive performance.

It is important to emphasize that this framework predicts EC numbers for a given enzyme but does not determine whether the input molecule is an enzyme. Future efforts should focus on benchmarking protein LLM models for this specific task.

Crucially, we show that protein LLM models should still be improved to further narrow or even surpass the performance gap with BLASTp and potentially become the gold standard in enzyme annotation. Overall, we strengthen the claim that BLASTp and LLM models are complementary tools that combined can be advantageous when employed in mainstream protein annotation routines and hard annotation tasks.

Supplementary Information

The online version contains supplementary material available at https://doi.org/10.1186/s12859-025-06081-9.

Supplementary file 1.

Supplementary file 2.

Acknowledgements
EMBL IT Support is acknowledged for the provision of computer and data storage servers.

Author Contributions
Capela J. developed the methodology and the analysis. Capela J., Zimmermann-Kogadeeva M., van Dijk A., de Ridder D., Dias O. and Rocha M. wrote the manuscript and conceptualized the study. Zimmermann-Kogadeeva M., van Dijk A., de Ridder D., Dias O. and Rocha M. supervised the study. All authors edited and approved the final manuscript.

Funding
FCT supported this study under the strategic funding of the UIDB/04469/2020 unit and by LABBELS (LA/P/0029/2020). We also thank FCT for the PhD fellowship to J. Capela (DFA/BD/08789/2021) and funding O. Dias (DOI 10.54499/CEEC-IND/03425/2018/CP1581/CT0020). J. Capela also acknowledge the European Molecular Biology Laboratory (EMBL) Corporate Partnership Programme for funding a short-term visitor fellowship. M. Zimmermann-Kogadeeva is supported by the EMBL.

Availibility of data and materials
Code for the model implementation and analysis is available at: https://github.com/jcapels/ec_numbers_prediction. P-values of all statistical tests can be found in the Supplementary Material.

Declarations

Ethics approval and consent to participate
Not applicable.

Consent for publication
Not applicable.

Competing interests
The authors declare that they have no Conflict of interest.

Received: 2 September 2024 Accepted: 11 February 2025
Published online: 27 February 2025

References

1. Keller MA, Piedrafita G, Ralser M. The widespread role of non-enzymatic reactions in cellular metabolism. Curr Opin Biotechnol. 2015;8(34):153–61.
2. Leveson-Gower RB, Mayer C, Roelfes G. The importance of catalytic promiscuity for enzyme design and evolution. Nat Rev Chem. 2019;3(12):687–705.
3. Ryu JY, Kim HU, Lee SY. Deep learning enables high-quality and high-throughput prediction of enzyme commission numbers. PNAS. 2019;116(28):13996–4001.
4. Kotera M, Okuno Y, Hattori M, et al. Computational assignment of the EC numbers for genomic-scale analysis of enzymatic reactions. JACS. 2004;126(50):16487–98.
5. Baldazzi D, Savojardo C, Martelli PL, et al. BENZ WS: the bologna ENZyme web server for four-level EC number annotation. Nucleic Acids Res. 2021;49(W1):W60–6.
6. Capela J, Lagoa D, Rodrigues R, et al. An improved framework for the reconstruction of high-quality genome-scale metabolic models. Nucleic Acids Res. 2022;50(11):6052–66.
7. Quester S, Schomburg D. EnzymeDetector: an integrated enzyme function prediction tool and database. BMC Bioinf. 2011;12(1):1–13.
8. Li Y, Wang S, Umarov R, Xie B, et al. DEEPre: sequence-based enzyme EC number prediction by deep learning. Bioinformatics. 2018;34(5):760–9.
9. Altschul SF, Gish W, Miller W, et al. Basic local alignment search tool. J Mol Biol. 1990;215(3):403–10.
10. Ferrari LD, Aitken S, van Hemert J, Goryanin I. EnzML: multi-label prediction of enzyme classes using InterPro signatures. BMC Bioinf. 2012;12(13):61.
11. Kumar N, Skolnick J. EFICAz2.5: application of a high-precision enzyme function predictor to 396 proteomes. Bioinformatics. 2012;28:2687–8.
12. Nagao C, Nagano N, Mizuguchi K. Prediction of detailed enzyme functions and identification of specificity determining residues by random forests. PLoS One. 2014;9(1): e84623.
13. Sanderson T, Bileschi ML, Belanger D, et al. ProteInfer deep neural networks for protein functional inference. eLife. 2023;12: e80942.
14. Shi Z, Deng R, Yuan Q, et al. Enzyme commission number prediction and benchmarking with hierarchical dual-core multitask learning framework. Research. 2023;1:6.

15. Buton N, Coste F, Cunff YL. Predicting enzymatic function of protein sequences with attention. Bioinformatics. 2023;10:39.
16. Yu T, Cui H, Li JC, et al. Enzyme function prediction using contrastive learning. Science. 2023;379:1358–63.
17. Kim GB, Kim JY, Lee JA, et al. Functional annotation of enzyme-encoding genes using deep learning with transformer layers. Nat Commun. 2023;11(14):7370.
18. Dalkiran A, Rifaioglu A, Martin MJ, et al. ECPred: a tool for the prediction of the enzymatic functions of protein sequences based on the EC nomenclature. BMC Bioinf. 2018;19:1–13.
19. Bateman A, Martin MJ, Orchard S, et al. UniProt: the universal protein knowledgebase in 2023. Nucleic Acids Res. 2023;1(51):D523–31.
20. Mistry J, Chuguransky S, Williams L, et al. Pfam: the protein families database in 2021. Nucleic Acids Res. 2021;49(D1):D412–9.
21. Schwartz AS, Hannum GJ, Dwiel ZR, et al. Deep semantic protein representation for annotation, discovery, and engineering. bioRxiv. 2018;.
22. Buchfink B, Xie C, Huson DH. Fast and sensitive protein alignment using DIAMOND. Nat Methods. 2015;12(1):59–60.
23. Rives A, Meier J, Sercu T, et al. Biological structure and function emerge from scaling unsupervised learning to 250 million protein sequences. Proc Natl Acad Sci. 2021;4:118.
24. Lin Z, Akin H, Rao R, et al. Evolutionary-scale prediction of atomic-level protein structure with a language model. Science. 2023;379(6637):1123–30.
25. Elnaggar A, Heinzinger M, Dallago C, et al. ProtTrans: toward understanding the language of life through self-supervised learning. IEEE Trans Pattern Anal Mach Intell. 2022;44(10):7112–27.
26. Fenoy E, Edera AA, Stegmayer G. Transfer learning in proteins: evaluating novel protein learned representations for bioinformatics tasks. Brief Bioinf. 2022;23(4):bbac232.
27. Steinegger M, Mirdita M, Söding J. Protein-level assembly increases protein sequence recovery from metagenomic samples many fold. Nature methods. 2019;16(7):603–6.
28. Wehrmann J, Cerri R, Barros R. Hierarchical multi-label classification networks. In: Dy J, Krause A, editors. Proc. 35th ICML. vol. 80 of Proc. Mach. Learn. Res.; 2018. p. 5075–5084.
29. Sechidis K, Tsoumakas G, Vlahavas I. On the stratification of multi-label data. In: Gunopulos D, Hofmann T, Malerba D, Vazirgiannis M, editors. Machine learning and knowledge discovery in databases. Berlin Heidelberg: Springer; 2011. p. 145–58.
30. Fu L, Niu B, Zhu Z, Wu S, Li W. CD-HIT: accelerated for clustering the next-generation sequencing data. Bioinformatics. 2012;28(23):3150–2.
31. Price MN, Wetmore KM, Waters RJ, et al. Mutant phenotypes for thousands of bacterial genes of unknown function. Nature. 2018;557(7706):503–9.
32. Choudhuri S. In: sequence alignment and similarity searching in genomic databases. Elsevier; 2014. p. 133–155.
33. Clark WT, Radivojac P. Information-theoretic evaluation of predicted ontological annotations. Bioinformatics. 2013;29(13):i53–61.
34. Piovesan D, Zago D, Joshi P, De Paolis Kaluza MC, Mehdiabadi M, Ramola R, et al. CAFA-evaluator: a python tool for benchmarking ontological classification methods. Bioinf Adv. 2024;4(1):vbae043.
35. Bileschi M, Belanger D, Bryant D, et al. Using deep learning to annotate the protein universe. Nat Biotech. 2022;40(6):932–7.
36. Crowe C, Molyneux S, Sharma SV, Zhang Y, Gkotsi DS, Connaris H, et al. Halogenases: a palette of emerging opportunities for synthetic biology-synthetic chemistry and C-H functionalisation. Chem Soc Rev. 2021;50(17):9443–81.