GigaScience, 2024, **13**, 1–17
DOI: 10.1093/gigascience/giae104
Technical Note

PlasGO: enhancing GO-based function prediction for plasmid-encoded proteins based on genetic structure

Yongxin Ji[1], Jiayu Shang[2], Jiaojiao Guan[1], Wei Zou[1], Herui Liao[1], Xubo Tang[1], and Yanni Sun[1,*]

[1]Department of Electrical Engineering, City University of Hong Kong, Kowloon, Hong Kong SAR (HKG), China
[2]Department of Information Engineering, The Chinese University of Hong Kong, Shatin, NT, Hong Kong SAR (HKG), China
*Correspondence address. Yanni Sun, Department of Electrical Engineering, City University of Hong Kong, 83 Tat Chee Avenue, Kowloon, Hong Kong (SAR), China. E-mail: yannisun@cityu.edu.hk

Abstract

Background: Plasmid, as a mobile genetic element, plays a pivotal role in facilitating the transfer of traits, such as antimicrobial resistance, among the bacterial community. Annotating plasmid-encoded proteins with the widely used Gene Ontology (GO) vocabulary is a fundamental step in various tasks, including plasmid mobility classification. However, GO prediction for plasmid-encoded proteins faces 2 major challenges: the high diversity of functions and the limited availability of high-quality GO annotations.

Results: In this study, we introduce PlasGO, a tool that leverages a hierarchical architecture to predict GO terms for plasmid proteins. PlasGO utilizes a powerful protein language model to learn the local context within protein sentences and a BERT model to capture the global context within plasmid sentences. Additionally, PlasGO allows users to control the precision by incorporating a self-attention confidence weighting mechanism. We rigorously evaluated PlasGO and benchmarked it against 7 state-of-the-art tools in a series of experiments. The experimental results collectively demonstrate that PlasGO has achieved commendable performance. PlasGO significantly expanded the annotations of the plasmid-encoded protein database by assigning high-confidence GO terms to over 95% of previously unannotated proteins, showcasing impressive precision of 0.8229, 0.7941, and 0.8870 for the 3 GO categories, respectively, as measured on the novel protein test set.

Conclusions: PlasGO, a hierarchical tool incorporating protein language models and BERT, significantly expanded plasmid protein annotations by predicting high-confidence GO terms. These annotations have been compiled into a database, which will serve as a valuable contribution to downstream plasmid analysis and research.

Keywords: GO term prediction, plasmid protein function, protein language model, BERT

Background

Plasmids are typically circular, extrachromosomal DNA molecules primarily found in bacteria. As a type of mobile genetic element (MGE), about half of them can mediate horizontal gene transfer (HGT) by transferring between different bacteria through a process known as conjugation [1, 2]. Consequently, the advantageous traits carried by the conjugative plasmids, such as antimicrobial resistance (AMR), will be disseminated among the bacterial community, thereby promoting the evolutionary adaptation of the host bacteria [3]. Plasmid-specific proteins can be classified into 2 main categories: core (backbone) proteins and accessory (payload) proteins [4]. The core proteins play essential roles in plasmids' housekeeping functions, encompassing replication, stability, and conjugation. They are instrumental in plasmid typing, such as replicon (Rep) typing, mobilization (MOB) typing, and mate–pair formation (MPF) typing [5]. Moreover, the core proteins exhibit a strong correlation with the plasmid host range, namely, the hosts to which the plasmid can be transferred or in which it can be maintained. On the other hand, the accessory proteins encode the host-beneficial traits, such as AMR and virulence. In summary, functional annotation of plasmid-encoded proteins not only helps identify the plasmid-carried traits but also provides valuable insights into predicting the transmission trajectory of these traits.

Gene Ontology (GO) is one of the most widely adopted vocabularies for describing protein functions, encompassing a collection of 42,255 GO terms by April 2024 [6]. The GO terms are structured in a directed acyclic graph (DAG) format, with three root terms representing the 3 GO categories: Molecular Function (MF), Biological Process (BP), and Cellular Component (CC). Therefore, a protein can be annotated with GO terms from all 3 aspects to achieve a comprehensive description of its functionality. According to the true path rule, if a protein is annotated with a specific GO term, it is also considered to be annotated with all the ancestor terms (semantically more general) of that particular GO term. Therefore, a protein can be annotated with more than 1 GO term, leading to the formulation of GO term prediction as a multilabel classification problem.

The rapid development of deep learning (DL) offers a promising approach, leveraging its strong generalization capability to predict the functions of novel proteins. Several DL methods have been proposed for GO term prediction utilizing only protein sequences as input. Among them, DeepSeq [7] and DeepGOPlus [8] are designed based on convolutional neural networks (CNNs). Specifically, DeepSeq first utilizes word embedding to represent each amino acid (AA) as a 23-dimensional vector and then feeds these embeddings to two 1-dimensional convolutional (Conv1d) layers to extract meaningful features for function prediction.

Received: July 4, 2024. **Revised:** October 29, 2024. **Accepted:** November 27, 2024

DeepGOPlus enhances the performance by incorporating 16 Conv1d layers with varying filter sizes, ranging from 8 to 128, in parallel. This enables the model to learn sequence motifs of different lengths, which play a crucial role in predicting protein function. The other 2 methods, TALE [9] and PFresGO [10], are built on the Transformer architecture and leverage the hierarchical information in the GO DAG. While using a Transformer layer to capture the dependencies among AAs within a protein, TALE also learns embeddings for the GO term labels to encode the ancestor relationship between GO terms. By multiplying the features learned by the Transformer and the label embeddings, a similarity matrix is generated for the final prediction, representing the similarity score between each AA and each label. PFresGO replaces the self-attention model in Transformer with a cross-attention mechanism, where GO terms act as the query to identify the functionally relevant AAs. Although there are a number of GO term prediction tools, they are not optimized for plasmid-encoded proteins.

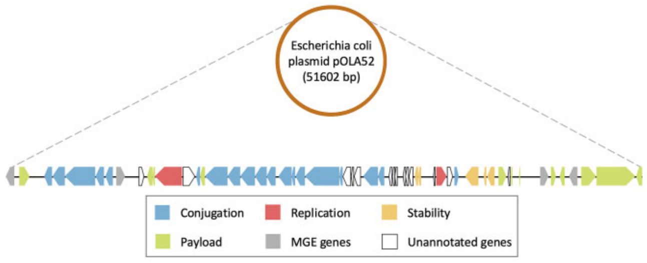


Figure 1: The flattened diagram of the circular plasmid pOLA52, which illustrates the CDS region of each gene, with the color representing the function of the encoded protein. The annotated encoded proteins were manually classified into 5 functional classes based on the gene product annotation in the NCBI database. In the diagram, a reversed pentagon block indicates that the corresponding CDS is located on the complementary strand.

Challenges for plasmid protein GO prediction

The prediction of GO terms for plasmid-encoded proteins presents 2 major challenges that have not been well addressed by generic GO prediction tools. First, compared to other types of biological entities, plasmids tend to encode a smaller number of proteins, but these proteins showcase a comparable level of functional diversity to more complicated peers. For instance, the manually reviewed Swiss-Prot database includes 323,202 proteins encoded in bacterial chromosomes, associated with 9,631 GO terms. In contrast, plasmid-encoded proteins, despite only accounting for nearly 1% of the protein count (3,202), are still associated with a considerable number of GO terms (3,318). This larger ratio of GO terms to protein count can be attributed to 2 main causes: the frequent genetic exchange events occurring between chromosomes and plasmids [11], as well as the gene flow between plasmids and phages mediated by phage–plasmid elements [12]. As a result, plasmids encode many proteins derived from both chromosomes and phages. Overall, the combination of a large number of GO terms (labels) and a relatively small number of proteins (training samples) increases the difficulty of the multilabel classification. The second challenge relates to the limited availability of high-quality GO annotations for plasmid-encoded proteins. For instance, considering the 678,197 nonredundant proteins encoded in 47,871 complete plasmids obtained from the RefSeq database, only 29.34% of these proteins possess annotated GO terms from at least 1 of the 3 GO categories. Alignment-based methods like the InterPro2GO pipeline [13] failed to extend the protein annotation rate due to their inability to identify signatures (protein families or domains) for the remaining uncharacterized proteins.

To address these challenges, we design a method that capitalizes on the genetic structures of plasmids for better GO prediction. Like human languages that possess a linguistic structure, genes residing on plasmids also exhibit a distinct biological structure, characterized by a modularization pattern [4]. This pattern often results in the division of a plasmid into functionally related segments, including areas dedicated to replication, conjugation, payload, and other plasmid-specific functions. The reason for this phenomenon is the dynamic evolution of plasmids over time, facilitated by the acquisition or loss of segments through recombination or the movement of MGEs [14]. Furthermore, functionally related segments within plasmids resemble phrases in human languages, attributable to both the similarity of protein functions and specific interactions between certain proteins within the same segment (e.g., relaxases interacting with type IV coupling proteins during conjugation). As depicted in Fig. 1, the large conjugative multidrug resistance plasmid pOLA52 [15] explicitly displays a modularization pattern, wherein the coding sequences (CDSs) with similar functions are more likely to be closely positioned. Interestingly, as shown on the right side of Fig. 1, several payload genes are interspersed with 2 transposases (a type of MGE gene), suggesting that this specific segment may have originated from HGT facilitated by transposons. Rapidly evolving language models have demonstrated significant advantages across various bioinformatics domains [16], including gene expression prediction [17], drug discovery [18], and tumor T-cell antigen (TTCA) classification [19]. Hence, as a major component of our methodology, we aim to leverage the language models to explore the modular structure of plasmids.

In addition, we will leverage protein language models (PLMs), which have demonstrated remarkable performance in various protein-related tasks by understanding the underlying semantic meaning of the language of life [20]. After self-supervised pretraining on a large corpus of protein sequences without manual labeling, foundation PLMs can generate protein embeddings that encapsulate learned knowledge. An important example of such knowledge is the correlation between 1-dimensional (1D) protein sequences and their corresponding 3-dimensional (3D) structures. The acquired prior biological knowledge can be leveraged to enhance protein function prediction through transfer learning. As an example, the ESM-2 family models [21] utilize a BERT-style [22] encoder variant with up to 15 billion parameters, making them the largest PLMs to date. After training with an unsupervised masked language modeling (MLM) objective on a dataset comprising over 60 million protein sequences, ESM-2 demonstrates superior performance in the classification of GO terms [23].

In this study, we have incorporated the above key observations and developed a dedicated tool called PlasGO for predicting GO terms of plasmid-encoded proteins. Rather than starting from scratch, we employ the state-of-the-art (SOTA) foundation PLM, ProtTrans [24], to generate biologically meaningful embedding for each plasmid-encoded protein as the raw input for our models. Then, we define plasmid sequences as a language using the vocabulary of proteins and leverage the powerful BERT model [22] to capture the genetic structures of plasmids. More specifically, we formulate the GO term prediction as a multilabel token classification task in natural language processing (NLP), where each protein is assigned 1 or more GO term labels. Additionally, to increase precision and filter high-confidence predictions, we integrate a self-attention confidence weighting mechanism to learn a confidence

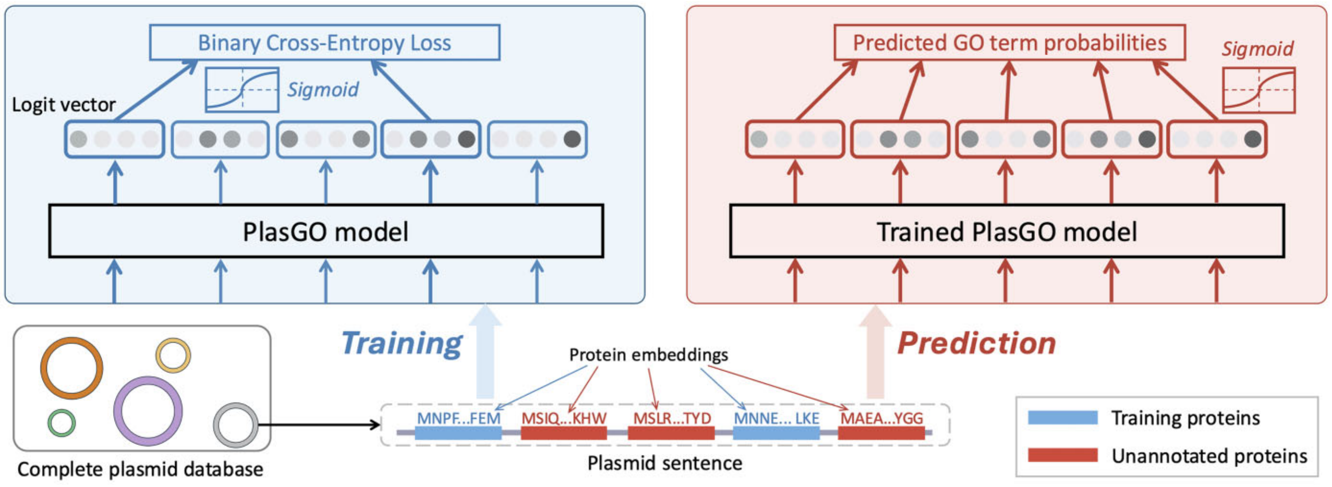


Figure 2: The workflows of PlasGO with a toy plasmid encoding 5 proteins as input during the training and prediction phases. The figure focuses on the token classification framework, emphasizing the learning of global context across different proteins, which are represented by the per-protein embeddings learned by PLM. Therefore, the toy plasmid is defined as a sentence comprising 5 protein embeddings, and the objective of PlasGO is to predict GO terms for each protein embedding within the plasmid sentence. Two of the 5 proteins (depicted in blue) have GO annotations and are included in the training set, while the remaining 3 (depicted in red) represent proteins without any GO annotation. Throughout both phases, PlasGO learns a vector (passed through sigmoid) for each protein, indicating the probabilities of its corresponding GO terms.

score for each predicted GO term. Thus, users can conduct automatic function annotation based on our predictions with associated confidence values. We rigorously evaluated PlasGO on the curated RefSeq dataset and a case study involving 2 well-studied conjugative plasmids. PlasGO consistently demonstrated superior performance across all experimental results, including the prediction of GO terms for novel proteins. We directly applied PlasGO to 678,197 proteins in the RefSeq plasmid sequences. PlasGO successfully extended high-confidence GO terms for over 95% of the unannotated proteins, which can provide important data for downstream plasmid research.

Material and Methods

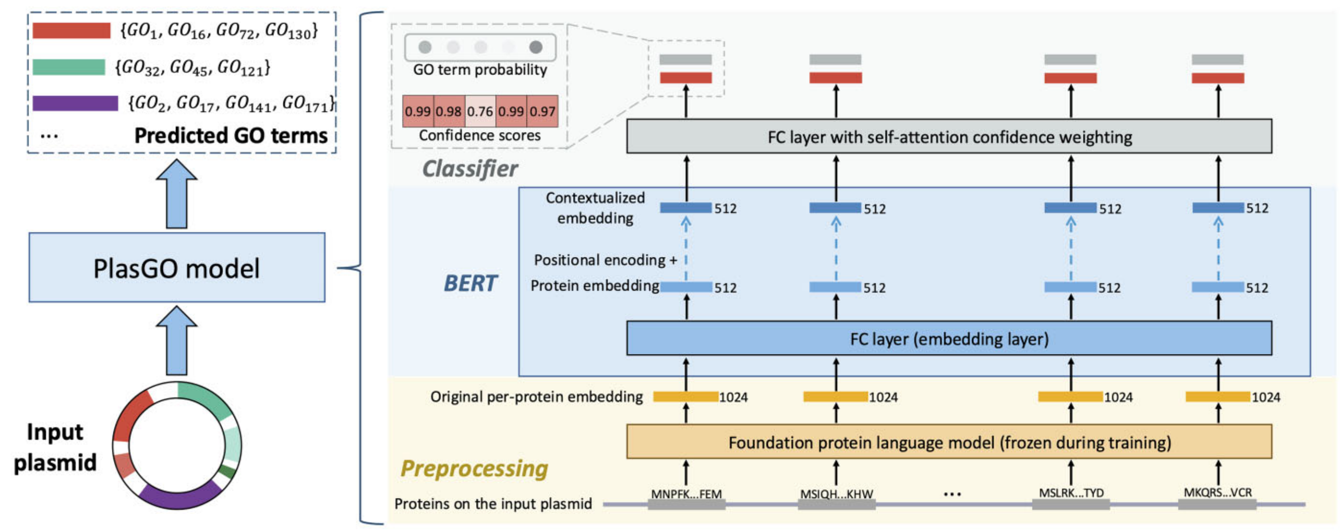
Design rationale

Based on the sequential features of protein sequences and the modular characteristics of plasmids, we designed a hierarchical architecture that considers both the local context within a protein sequence and the global context across different proteins. Following the language analogy in NLP, we establish 2 types of sentences: plasmid sentences (global), in which encoded proteins serve as tokens, and protein sentences (local), where individual AAs act as tokens. Given that a series of powerful PLMs [21, 24, 25] have demonstrated exceptional performance in extracting biologically meaningful embeddings for protein sentences, our primary contribution lies in the global learning of plasmid sentences. As shown in Fig. 2, we frame the GO term prediction for plasmid-encoded proteins as a token classification task in NLP, where GO term labels are assigned to individual proteins within a plasmid sentence. Correspondingly, we leverage the BERT model as the central component of PlasGO to capture the structure of plasmid sentences.

The method design of PlasGO incorporates 2 distinctive features. First, because a protein can be annotated with multiple GO terms, PlasGO is designed as a multiclass, multilabel token classification approach. In other words, the BERT model predicts multiple labels for each protein, which sets it apart from traditional token classification tasks such as named entity recognition (NER) [26]. Second, PlasGO accepts the same plasmid as input for both supervised training and prediction while focusing on different proteins in each phase. The workflows are depicted in Fig. 2 using a toy plasmid. During training, binary cross-entropy (BCE) loss is calculated by comparing the learned GO term probability vectors of the training proteins against their actual GO annotations. Masking is exclusively applied to proteins that are not in the training set (e.g., the second, third, and fifth proteins in Fig. 2) when computing the loss function given their lack of labels hindering their involvement in backpropagation. In the prediction phase, the trained PlasGO model generates prediction results for all unannotated proteins. Notably, throughout both training and prediction, embeddings for all proteins encoded on a plasmid will be input into the PlasGO model, ensuring no loss of informative genomic context during each phase. In essence, the central concept behind PlasGO is to enhance the function prediction of unannotated plasmid-encoded proteins by leveraging plasmid-level contextual information. On the other hand, even in scenarios where all nearby proteins of a particular protein lack annotations, PlasGO still works by directly predicting GO terms for that protein using the local PLM embeddings.

Overview of the PlasGO model

As shown in Fig. 3, the PlasGO model takes as input a sentence representation of the plasmid, wherein translated proteins are arranged in the same order as their encoding in the plasmid. Consequently, the PlasGO model generates high-confidence predicted GO terms for each protein as its output. Specifically, the PlasGO model consists of 3 modules executed linearly, with the output of each module feeding into the next. The first is the preprocessing module, responsible for generating original per-protein embeddings. This is accomplished by utilizing a pretrained foundation PLM, which extracts biophysical features of the input protein sequences. The second module is a BERT model. By capturing the contextual information at the plasmid sentence level, the BERT model enhances the original embeddings, encompassing a deeper understanding of the functional characteristics of the proteins. The final is the classifier module, designed explicitly for multilabel token classification. Within this module, we employ a combina-



$\{GO_1, GO_{16}, GO_{72}, GO_{130}\}$
$\{GO_{32}, GO_{45}, GO_{121}\}$
$\{GO_2, GO_{17}, GO_{141}, GO_{171}\}$
Predicted GO terms
PlasGO model
Input plasmid
GO term probability
0.99 0.98 0.76 0.99 0.97
Confidence scores
Classifier
FC layer with self-attention confidence weighting
Contextualized embedding
512
BERT
Positional encoding +
Protein embedding
FC layer (embedding layer)
Original per-protein embedding
1024
Foundation protein language model (frozen during training)
Preprocessing
Proteins on the input plasmid
MNPFK...FEM
MSIQH...KHW
MSLRK...TYD
MKQRS...VCR

Figure 3: The pipeline of the PlasGO model. The PlasGO model takes as input a series of proteins encoded in a plasmid, and its output comprises high-confidence GO term annotations in nominal format for these proteins. On the right side, the 3 main modules of the PlasGO model—namely, preprocessing, BERT, and classifier—are displayed in a bottom-to-top arrangement. In the preprocessing stage, the proteins (represented as multiple gray bars) are organized in the same order as their encoding in the plasmid. These proteins are then fed into the foundation PLM to extract biologically meaningful embeddings (represented by yellow bars). Next, in the BERT module, an FC layer is utilized to transform the original embeddings learned by PLM into protein embeddings (represented by light blue bars). Additionally, multiple Transformer encoders are employed to capture the global context between protein embeddings. Lastly, using the learned contextualized embeddings (represented by deep blue bars), the classifier predicts a GO term probability vector (represented by a light gray bar) for each protein. Simultaneously, a corresponding confidence score vector (represented by a red bar) of the same dimension is also generated. Only the GO terms with both high predicted probabilities and high confidence scores are retained as nominal-format GO term annotations for the respective protein.

tion of a fully connected (FC) layer and a self-attention confidence weighting mechanism. As a result, for each protein, the classifier generates a GO term probability vector along with a confidence score vector of the same dimension. By removing predictions with low confidence scores, the PlasGO model achieves enhanced accuracy in predicting the GO terms. In the subsequent sections, we will provide a more detailed description of the 3 modules.

Extract original per-protein embedding with foundation PLM

In this section, we utilize ProtTrans to generate embeddings for each plasmid-encoded protein. Drawing inspiration from concepts in NLP, ProtTrans considers entire proteins as sentences, where individual AAs are analogous to words (tokens). As depicted in Fig. 4, the input toy protein sequence "MNPF" is initially tokenized into an array of individual AA tokens [M, N, P, F]. Subsequently, all the AA tokens undergo an embedding layer incorporating positional encoding. This will convert the AA tokens into high-dimensional vectors (1,024 dimensions). The resulting embedded vectors are then fed into an *L*-layer Transformer encoder, which captures the semantic meaning of individual AA tokens and their contextual relationships within the protein sequence. In the final step, the output of the last encoder layer for each AA token (per-residue embedding) is concatenated and pooled using a global average pooling (GAP) operation [27]. This involves taking the average of each per-residue embedding, resulting in the final per-protein embedding.

ProtTrans provides several pretrained models with different parameter numbers and architectures. We selected ProtT5-XL-U50 for PlasGO due to its superior performance across various downstream prediction tasks, as demonstrated in the study by Elnaggar et al. [24]. ProtT5-XL-U50 is built upon the T5-3B model architecture [28] and was trained using the MLM approach [22] on the UniRef50 dataset, which contains 45 million protein sequences. In

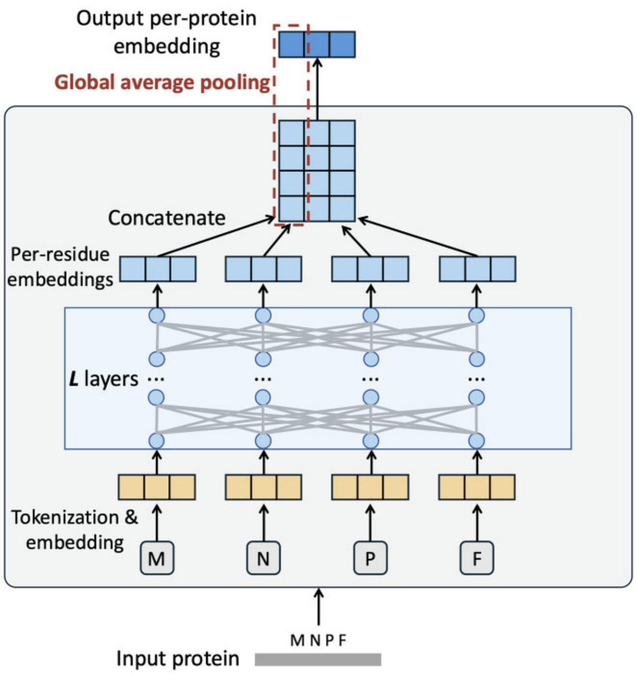


Figure 4: An overview on how the ProtTrans model generates the per-protein embeddings for input plasmid-encoded proteins.

this work, we employ the ProtTrans model in a feature extraction manner rather than fine-tuning it, which means the per-protein embedding extraction process can be considered a preprocessing step. Prior to both training and prediction, all the involved proteins are input into ProtTrans, and the resulting per-protein embeddings are saved for later utilization.

Capture contextual information on plasmids with BERT

We implement our BERT module following the standard BERT architecture [22] with 3 modifications to accommodate our method design. First, we exclude the 2 tokens “[CLS]” and “[SEP]” defined in standard BERT, as they are not utilized for token classification. Second, as mentioned earlier, our BERT module is primarily designed to capture functionally related segments in plasmids, similar to how phrases are learned in NLP. Thus, a limited number of layers (Transformer encoders) L is adequate for capturing “phrase”-level information in plasmid sentences [29]. Specifically, we set $L = 4$ for the MF and BP categories and $L = 2$ for the CC category, considering the relatively smaller dataset size and label size for CC. Third, we remove the token embedding layer in the standard BERT, which maps token indices to vectors, as we do not represent proteins in an indexed form. Instead, in the previous module, we have preprocessed the embedding step using ProtTrans, where an original embedding is extracted for each sequential protein (AA sequence). Correspondingly, we incorporate an FC layer as the initial layer of our BERT module, which is trained to transform the original embeddings into function-related protein embeddings. Next, position embeddings are introduced and combined with the protein embeddings:

$$\begin{cases} E_{pro} = FC(E_o, W_{FC}) \\ \tilde{E}_p = Embed(E_p, W_{E_p}) \\ X = E_{pro} + \tilde{E}_p \end{cases} \tag{1}$$

In this work, we set the maximum length of proteins in a plasmid sentence to 56, as it represents the median length observed in our curated database of complete plasmids. For sentences with fewer than 56 proteins, the “[PAD]” tokens will be padded at the end of the sentences, and they will be masked during the training. In addition, we utilize a hidden size of $H = 512$ and a number of self-attention heads of $A = 8$ for the Transformer encoders. Therefore, $E_o \in \mathbb{R}^{56\times1024}$ is the original embeddings extracted by ProtTrans for the 56 proteins, while $E_p \in \mathbb{R}^{56\times1}$ is the position index vector. $W_{FC} \in \mathbb{R}^{1024\times512}$ and $W_{E_p} \in \mathbb{R}^{56\times512}$ are learnable weight matrices used for the linear transformation of protein embeddings and position embeddings, respectively. As a result, both the protein embeddings E_{pro} and the position embeddings $\tilde{E}_p$ have a dimension of $\mathbb{R}^{56\times512}$, and their sum, denoted as X, will be fed into the subsequent Transformer encoders.

Learn confidence scores for multilabel token classification

In current databases, it is inevitable to encounter incomplete GO annotations because some proteins are annotated with GO terms at varying levels of specificity [30]. Furthermore, accurately predicting certain GO terms poses challenges due to the inherent complexities involved in capturing functional factors such as protein domains or AAs. Consequently, learning a confidence score for each prediction is advantageous as it enables the rejection of uncertain predictions. Inspired by the self-cure network introduced by Wang et al. [31], we integrate a self-attention confidence weighting mechanism tailored for multilabel classification within our classifier module. The detailed architecture of the classifier module is depicted in Fig. 5. The contextualized embeddings learned from the BERT module for each protein will be fed in parallel to the classifier module. Instead of utilizing a single FC layer for multilabel classification, our classifier module incorporates 2 branches: one for learning logits and another for learning confidence scores. Prior to the final prediction using sigmoid, the logits will undergo attention-weighting based on the learned confidence scores. Intuitively, during training, the model will inherently learn to assign smaller confidence weights to incorrect predictions in order to minimize the loss resulting from these inaccuracies. On the contrary, the true predictions tend to receive larger confidence weights for the logits. The final step involves using both the confidence scores and weighted predicted probabilities to determine the nominal-format GO term predictions, retaining only those predictions characterized by high values in both aspects. Notably, as a modification to the original architecture proposed in [31], we adapt it for the multilabel classification task of GO term prediction by learning a confidence score for each prediction instead of each sample.

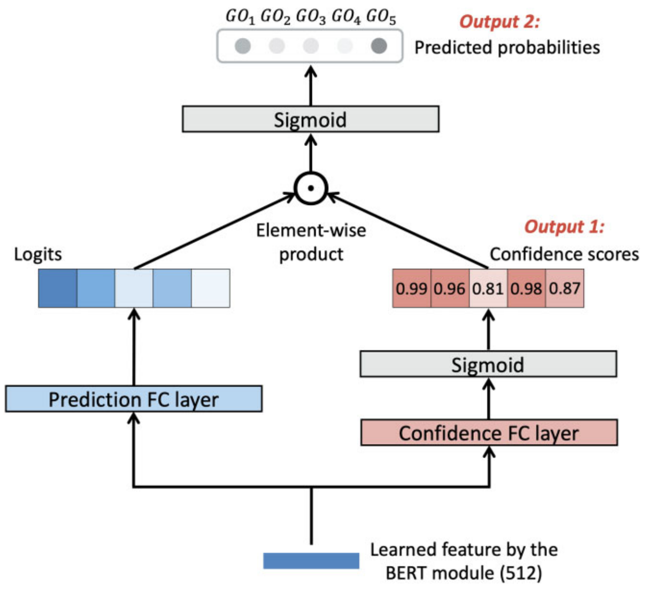


Figure 5: The model architecture of the classifier module incorporating the self-attention confidence weighting mechanism. We utilize a toy label set comprising 5 GO terms for illustration. The figure displays the 2 branches, with one dedicated to learning logits and the other focused on learning confidence scores, positioned on the left and right sides, respectively. The attention weighting is implemented between the outputs of the 2 branches to obtain the final multilabel prediction. Both the confidence score vector and the final predicted probability vector will be output by the classifier module to determine the high-confidence GO term predictions in nominal format.

Specifically, the input feature will be forwarded through 2 FC layers with identical sizes but distinct parameters. These layers consist of a prediction FC layer, responsible for learning logits, and a confidence FC layer, specifically designed to get confidence scores using a sigmoid function. Following that, the confidence scores will serve as the attention weights for the logits through an element-wise dot product:

$$\begin{cases} L = FC(F, W_{FC_p}) \\ C = \sigma(FC(F, W_{FC_c})) \\ P = \sigma(L \odot C) \end{cases} \tag{2}$$

where $F \in \mathbb{R}^{56\times512}$ is the input features of the 56 proteins within 1 sentence. Here, n represents the number of GO terms, and σ is the sigmoid function. Accordingly, W_{FC_p} and W_{FC_c} represent the learnable weight matrices, both having dimensions of $\mathbb{R}^{512\times n}$, associated with the prediction FC layer and the confidence FC layer, respectively. Additionally, L and C denote the learned logits and confidence score matrix, respectively, both having dimensions of $\mathbb{R}^{56\times n}$. The final predicted probability matrix $P \in \mathbb{R}^{56\times n}$ is obtained by element-wise multiplying ($\odot$) the matrices L and C, followed by normalization using the sigmoid function. For the multilabel token classification of PlasGO, we define the logit-weighted binary cross-entropy loss (WBCE Loss) as follows:

$$\mathcal{L}_{\text{WBCE}} = -\frac{1}{N}\sum_{i=1}^{N} y_i \cdot log(\tilde{P}_i) + (1 - y_i) \cdot log(1 - \tilde{P}_i) \qquad (3)$$

where N denotes the total number of predictions, which is equal to $56 \times n$ for a single sentence. $\tilde{P}$ represents the flattened format of the probability matrix P, with a dimension of $1 \times N$. y_i represents the true label of the corresponding GO term, where it takes the value 1 if the protein is annotated with that particular GO term and 0 otherwise.

To further promote the ability of the model to distinguish low-confidence and high-confidence predictions, a rank regularization loss (RR Loss) is incorporated into the total loss function: $\mathcal{L}_{total} = \mathcal{L}_{\text{WBCE}} + \mathcal{L}_{\text{RR}}$. The calculation of RR Loss and the selection of high-confidence GO term predictions in nominal format are detailed in Supplementary Section S1.

Data curation and model training

RefSeq dataset

We chose to conduct our experiments using RefSeq due to the rigorous quality assurance (QA) checks administered by NCBI staff on all data within RefSeq before its public release [32]. This meticulous process results in high-quality RefSeq protein databases that significantly mitigate the occurrence of incorrect GO annotations. Moreover, the RefSeq database provides access to the genomic context information, specifically the order in which proteins are encoded in the plasmid, a key feature for PlasGO that is not available in other protein-only databases like Swiss-Prot. On the other hand, despite the long-standing issue of incompleteness in the GO annotation domain, several previous studies have demonstrated the meaningfulness and reliability of current large-scale GO evaluations [33, 34]. As GO annotations continue to expand over time, the problem of incompleteness is expected to gradually improve. In light of these considerations, we opted for the use of the more accurate RefSeq database in developing our tool, as it allows our model to better capture the distinctive features of plasmid-encoded proteins with reduced noise and misinterpretation.

We initially downloaded all available plasmids from the NCBI RefSeq plasmid database [35], along with their corresponding protein sequences translated from CDSs, excluding pseudogenes. They can be stored in a dictionary format $plasmid_A$: $[protein_{A1}, protein_{A2}, \ldots]$, where the keys are plasmids and the values are lists of proteins arranged in the order they are encoded in the respective plasmids. The focus of this work is proteins in regular plasmids. Thus, we only keep plasmids with lengths between 1K and 350K to ensure each plasmid has at least 1 encoded protein and no megaplasmids are included [36]. We restricted the maximum protein length to 1 Kbp for training PlasGO because this limit is computationally efficient for the Transformer architecture, a common practice followed by many state-of-the-art protein-related methods such as ESM [37] and PFresGO [10]. Nonetheless, in the prediction phase or when utilizing our PlasGO tool, no length restrictions are imposed. We evaluated the generalization of PlasGO to proteins larger than 1 Kbp in Supplementary Section S2. Finally, the curated proteins' associated GO terms were also retrieved from the RefSeq database, if available.

Since the GO terms provided in the database are commonly more specific (on the lower levels), we augment the original annotations by propagating all ancestors of the GO terms to their corresponding proteins. As a consequence, the resulting list describing 1 protein's function often includes a large number of highly redundant GO terms, which reduces interpretability for users. Several tools have been proposed to tackle this problem by clustering GO terms with high semantic similarity. The measurement of semantic similarity [38] relies on the proximity of 2 GO terms within the GO DAG. When 2 GO terms exhibit greater semantic similarity, they tend to be more functionally related. Among these tools, we selected the widely adopted REVIGO [39] to group 1,602 original GO terms into 487 clusters. If at least 1 GO term within a cluster is annotated to a protein, then that cluster will be assigned to the protein. Importantly, the GO term label clustering strategy is applied to the retraining processes of all benchmarked tools, ensuring a fair comparison of their performance. Afterward, we deduplicated the proteins by clustering protein sequences using MMseqs2 [40], considering pairs with at least 90% identity and 80% overlap, and only keeping the longest protein from each cluster. In line with the UniRef database [41], the annotations of the kept protein are determined as the intersection of the annotations of all members within the protein cluster. As a result, these 2 clustering processes have partially corrected misannotations of GO terms in the database.

Consistent with the settings commonly used in GO term prediction tools, we selected the GO term clusters associated with $\geq$50 proteins (excluding the 3 root terms) as the labels for our curated dataset. This resulted in 172, 174, and 31 cluster-level labels for the MF, BP, and CC categories, respectively. Subsequently, we devised a protein-based data splitting strategy to simulate the real scenario where plasmid sequence data are available, yet a majority of the encoded proteins lack annotations. For each GO category, we allocated 10% of the most recently released proteins with GO annotations as the test set. Additionally, we ensure that the novel test set significantly differs from the training set in terms of protein sequences. Therefore, among the remaining 90% annotated proteins, those lacking significant alignments (E-value>1e-3) to the test set were assigned to the training set, while others were assigned to the validation set. This splitting strategy poses significant challenges for both PlasGO and other DL methods. The final step involves converting plasmids into sentences for PlasGO's training and prediction. Plasmids containing more than 56 proteins were divided into multiple segments with an overlap of 14 proteins (1/4 of the maximum length). The curated dataset will be utilized for retraining all benchmarked tools, and a comprehensive overview of its specific details can be found in Table 1. Furthermore, experiments conducted using a plasmid-based data splitting strategy, including 4 groups of leave-one-genus-out benchmarking experiments and a 5-fold cross-validation, are detailed in Supplementary Sections S9 and S10, respectively.

PlasGO model training

We trained 3 PlasGO models, each specifically designed for 1 of the 3 GO categories. The models were trained with a batch size of 32 and a learning rate of 1e-4. Due to the smaller data size and increased susceptibility to overfitting in the CC category, we applied a dropout rate of 0.2 for CC and 0.1 for MF and BP. Additionally, we incorporated a warmup strategy by allocating 5% of the total training steps to gradually increase the learning rate from a small value to 1e-4. The methods employed to prevent overfitting in the PlasGO model, including dropout, model simplification, regularization, early stopping, and cross-validation, are outlined in Supplementary Section S3. Subsequently, the learning rate was

Table 1: The specific information of the curated dataset. In this table, the term "sentence" specifically refers to the plasmid sentence, which is composed of multiple proteins. Besides, the "# of sentences" item in the last 3 columns represents the count of sentences that contain at least 1 protein from the respective protein set

GO category	# of sentences (# of plasmids)	# of deduplicated proteins (# of original proteins)	# of annotated proteins	Training set size (# of sentences)	Validation set size (# of sentences)	Test set size (# of sentences)
MF	89,835 (47,871)	678,197 (1,103,790)	173,666	99,806 (87,198)	56,491 (44,026)	17,369 (77,074)
BP			99,945	60,143 (81,068)	29,768 (72,128)	10,034 (32,751)
CC			28,081	21,228 (77,877)	4,045 (18,226)	2,808 (8,952)

linearly decayed to enhance generalization and expedite convergence. Using the NVIDIA GeForce RTX 3090 Blower 24G graphics card, each model underwent 10 epochs of training, with approximate durations of 65 minutes for MF, 59 minutes for BP, and 51 minutes for CC. A more detailed discussion of the computational costs and resource requirements for training and running PlasGO can be found in Supplementary Section S4. Moreover, inspired by the iterative alignment tool PSI-BLAST [42], we developed a post-training phase—a fine-tuning strategy that iteratively refines the model with high-confidence pseudo-labeling. The methodology is elaborated in Supplementary Section S5.

Results

Experimental setup

Metrics

In our benchmark experiments, we assess the performance using 2 commonly used metrics in the CAFA challenge [34]: the protein-centric F_{max}, which measures the accuracy of assigning GO terms to a protein, and the term-centric area under the precision–recall curve (AUPR), which evaluates the accuracy of predicting which proteins are associated with a given GO term. In other words, we first calculate F_{max} for each protein and AUPR for each GO term separately. Subsequently, we obtain the average of these individual metrics as an overall performance evaluation. Importantly, dataset imbalance can lead to a high F_{max} but a low AUPR if the tool consistently fails to predict certain low-frequency GO term labels. Therefore, utilizing both metrics ensures a more accurate and comprehensive assessment of the tools' performance in GO term prediction.

F_{max} is defined as the highest F_1-score achieved among all probability cutoffs θ. To elaborate, we calculate the F_1-score for each θ value ranging from 0 to 1, using a stride of 0.01, and select the maximum score as F_{max}:

$$Precision_i(\theta) = \frac{\text{Correctly Predicted Terms } (i, \theta)}{\text{Predicted Terms } (i, \theta)} \quad (4)$$

$$Recall_i(\theta) = \frac{\text{Correctly Predicted Terms } (i, \theta)}{\text{True Terms } (i)} \quad (5)$$

$$AvgP(\theta) = \frac{1}{m(\theta)} \cdot \sum_{i=1}^{m(\theta)} Precision_i(\theta) \quad (6)$$

$$AvgR(\theta) = \frac{1}{n} \cdot \sum_{i=1}^{n} Recall_i(\theta) \quad (7)$$

$$F_{max} = \max_{\theta} \{F_1(\theta)\} = \max_{\theta} \left\{ \frac{2 \cdot AvgP(\theta) \cdot AvgR(\theta)}{AvgP(\theta) + AvgR(\theta)} \right\} \quad (8)$$

where $Precision_i(\theta)$ and $Recall_i(\theta)$ represent the precision and recall values for protein i with the cutoff θ. Correspondingly, $AvgP(\theta)$ and $AvgR(\theta)$ represent the average precision and recall values calculated for proteins with at least 1 predicted GO term ($m(\theta)$) and all proteins (n), respectively. *Predicted Terms* (i, θ) denotes the number of GO terms for protein i that have a predicted probability greater than θ. *True Terms* (i) represents the number of GO terms that are truly annotated for protein i. Empirically, the F_{max} metric reaches its peak when the cutoff θ is around 0.3, prompting many tools to utilize 0.3 as the default probability threshold for GO term prediction. We provide further elucidation on the rationale behind the F_{max} metric in Supplementary Section S6.

SOTA tools for benchmarking

We compared PlasGO with 7 deep learning–based tools that can be used for protein function annotation. These tools encompass DeepGOPlus [8], PFresGO [10], DeepSeq [7], TALE [9], ESM-2 [21], a codon-based large language model (CodonBERT [43]), and an approach trained for accurate protein structure alignments (TM-Vec [44]). In particular, we selected the model *esm2_t36_3B_UR50D* with 36 Transformer layers from the ESM-2 family models to maintain consistency with the ProtT5 model employed for PlasGO, both of which consist of 3 billion parameters. Due to our dataset splitting strategy, which resulted in no significant alignment between the test set and training set, we did not include any sequence alignment tools such as Diamond [45] in our comparison. Additionally, gLM [46], a pretrained genomic language model that combines PLM and BERT family models, is trained on millions of metagenomic scaffolds. It was not included in the benchmarking due to the contextualized embeddings generated by gLM showing a weak correlation with GO annotations for plasmid-encoded proteins. A more in-depth discussion comparing PlasGO and gLM can be found in Supplementary Section S7. Notably, all 7 selected SOTA tools can be optimized for GO term prediction for plasmid-encoded proteins using the same curated RefSeq dataset as PlasGO, ensuring a fair and consistent comparison of algorithms. To be specific, we performed model retraining for the first 4 tools, and in the case of ESM-2 and CodonBERT, we trained a classifier utilizing its learned embeddings. Besides, for TM-Vec, we created a custom database using our curated training set. There are 2 main reasons for this optimization process. First, some of the labels determined by us do not exist in their default models, making them unable to predict those labels. For example, the default PFresGO can only predict 125 out of 172 (72.67%) MF labels, 141 out of 174 (81.03%) BP labels, and 21 out of 31 (67.74%) CC labels from our label set. Second, the proteins used to train their default models span across various organisms and have less emphasis on plasmids. As a result, their default models exhibit inferior performance compared to our retrained models. For instance, among the subset of labels that can be predicted by the default PFresGO, it achieved F_{max} scores of 0.6514 and 0.6226 for MF and BP, respectively, while our retrained PFresGO improved significantly to 0.7039 and 0.7189, respectively.

When conducting benchmarking with ESM-2 and CodonBERT, we initially extracted the embeddings for each protein and then

Table 2: The performance of different classification methods using ProtT5 embeddings as input on the RefSeq test set. The first method involves training a 3-layer DNN using ProtT5 embeddings as input, utilizing the same curated RefSeq dataset aligned with PlasGO. The second method deviates from the standard PlasGO approach solely during the prediction phase. Specifically, a single test protein is treated as a test sentence with a length of 1, which is then inputted into the PlasGO model. Consequently, the attention mechanisms are disabled for the second method.

Method	GO category	F_{max}	AUPR
3-layer DNN classifier	MF	0.7662	0.4550
	BP	0.7498	0.3903
	CC	0.7778	0.4639
PlasGO (single test protein)	MF	0.7808	0.4950
	BP	0.7512	0.4088
	CC	0.7992	0.4406
PlasGO (standard)	MF	0.8070	0.5165
	BP	0.7855	0.4638
	CC	0.7926	0.5109

trained a deep neural network (DNN) as their GO term prediction classifier. In addition, TM-Vec generates structure alignments in a format similar to TM-align [47], represented as a triad $\{q, t, tmscore(q, t)\}$. Here, q denotes the query protein from the test set, t represents the target protein from the training set database, and $tmscore(q, t)$ corresponds to the template modeling score (TM-score) of the alignment. Alignments with a $tmscore(q, t)$ below 0.5, indicating a low structural similarity [48], are excluded from further analysis. Inspired by the DiamondScore method proposed by Kulmanov et al. [8], we compute the GO term probability vector for TM-Vec using the following formula:

$$P(q, f) = \frac{\sum_{t\in T} tmscore(q, t) * I(f \in F_t)}{\sum_{t\in T} tmscore(q, t)} \quad (9)$$

where $P(q, f)$ is the predicted probability that the query protein q is annotated with the GO term f. T represents the set of target proteins found in the significant alignments of the query protein q. $I(f \in F_t)$ is the identity function that returns 1 if the GO term f is present in the true annotations F_t of the target protein t and 0 otherwise.

Ablation studies: validating PlasGO's design rationale

Evaluation of BERT module in PlasGO

We first conducted an ablation study to investigate whether the BERT module effectively captures the contextual information on plasmids and improves the GO term prediction. Specifically, we compared the performance of 3 different classification methods, all utilizing protein embeddings from ProtT5 as input. The first method serves as the baseline, entailing the training of a 3-layer DNN classifier for each GO category using the identical dataset as PlasGO. This method does not leverage any plasmid-level contextual information as it predicts each protein independently. The second method involves using PlasGO for prediction by inputting a single test protein. In other words, each testing sentence has a length of 1, with 55 "[PAD]" tokens padded at the end of each test protein. In this approach, the proteins are also predicted individually. Thus, the main effective component is the embedding layer, while the attention blocks remain inactive. The third method is the standard PlasGO, which fully incorporates our design rationale. The experimental results are shown in Table 2. The standard PlasGO achieved the best performance on all the GO categories, indicating that the contextual information captured by the BERT module actively contributes to improving the GO term prediction. Additionally, the second method has a better performance compared to the DNN classifier. This suggests that despite the inactive attention blocks, the embedding layer still captured partial contextual information.

Evaluation of the foundation PLMs

Given that the protein embeddings extracted by the foundation PLM are the sole input to PlasGO's BERT module, it is critical to ensure that the input embeddings are highly informative. To this end, we evaluated the performance of PlasGO by utilizing embeddings extracted from 4 distinct ProtTrans models, which were constructed based on 4 prominent language models in NLP: T5 [28], BERT [22], XLNet [49], and Albert [50]. The results align closely with the per-protein prediction tasks reported in ProtTrans's study [24] (Table 3). PlasGO achieved the best results using ProtT5, while the differences in performance between PlasGO using other PLMs were minimal. This suggests that the model size of ProtT5 (3B) is optimal for learning the most informative embeddings from the vast number of training proteins (e.g., 45M in the UniRef50 database). Thus, we select ProtT5 to extract the original per-protein embeddings for PlasGO.

Performance on the RefSeq test set

We compared PlasGO with the other 7 tools on our curated RefSeq test set. During the evaluation of PlasGO, if a protein appears in multiple testing sentences, its final probability vector is determined by averaging all its predictions across these sentences. The predicted results of all the tools are shown in Fig. 6. Additional statistical significance test (the nonparametric McNemar's test) results can be found in Supplementary Section S8. Overall, PlasGO attained the highest scores for both F_{max} and AUPR on all 3 GO categories. A closer looks show that the subsequent top-performing tools (PFresGO, ESM-2, TM-Vec) all leverage the pretrained PLMs to encode protein-level or residue-level features for protein sequences. This observation suggests that the inherent relationships between AAs captured by PLMs can substantially enhance the prediction of GO terms. This aligns with the finding that many proteins execute their functions through spatially aggregated clusters of critical residues [51]. Specifically, PFresGO and TM-Vec utilize residue-level features, while PlasGO leverages protein-level features, resulting in significant time and memory savings and also scalability to a higher number of proteins. This advantage is achieved by implementing the BERT model to learn from plasmid-level corpora.

Although CodonBERT is also a large language model, it did not surpass the top 4 tools utilizing PLMs. This suggests that training on codon-level tokens does not offer the same enhancements as training on amino acid–level tokens for function prediction tasks. Nevertheless, this observation does not negate the superior advantages of codon-based models in specific tasks such as protein expression prediction or protein abundance prediction [23, 43]. Additionally, DeepGOPlus achieves the highest performance among the tools trained from scratch. However, it is still challenging for them to compete with the tools trained based on pretrained PLMs.

To further evaluate PlasGO's capability to generalize to novel proteins, including those from plasmid taxonomies not present in our training data, we performed 4 groups of leave-one-genus-out benchmarking experiments and a 5-fold cross-validation, all

Table 3: The performance of PlasGO trained with different foundation PLMs on the RefSeq test set

Pre-trained PLM	GO category	F_{max}	AUPR	Parameters
ProtT5	MF	0.8070	0.5165	3B
	BP	0.7855	0.4638	
	CC	0.7926	0.5109	
ProtBert	MF	0.7462	0.4532	420M
	BP	0.7447	0.3813	
	CC	0.7444	0.3915	
ProtXLNet	MF	0.7396	0.4693	409M
	BP	0.7526	0.3622	
	CC	0.7673	0.4567	
ProtAlbert	MF	0.7541	0.4220	224M
	BP	0.7396	0.3503	
	CC	0.7679	0.4393	

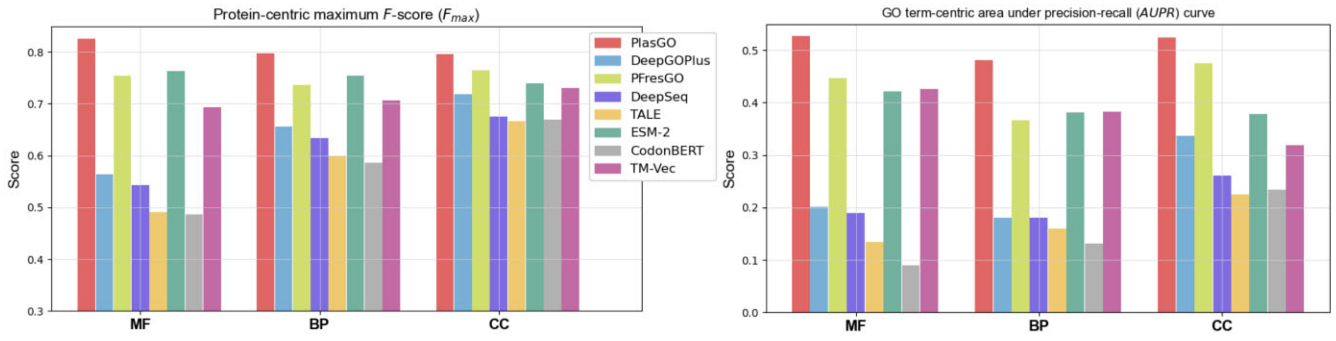


Figure 6: The performance of different tools on the RefSeq test set assessed based on two metrics, (A) F_{max} and (B) AUPR, and measured across the 3 GO categories.

with plasmid-based data splitting. The results are included in Supplementary Sections S9 and S10, respectively.

Visualization of the PlasGO embeddings

A visual comparison between the original embeddings generated by ProtTrans and the contextualized embeddings learned by the BERT module of PlasGO can provide valuable insights into how the design of the BERT module contributes to enhancing GO term prediction. Given that the multilabel GO term prediction can be considered as many individual binary classifications, we conducted visualization experiments focusing on several GO terms that are highly related to plasmid-specific functions. Specifically, we first selected a few representative plasmid-specific proteins, encompassing both core proteins and accessory proteins. Subsequently, we obtained the corresponding GO annotations for these proteins from the Swiss-Prot database. On one hand, we curated a subset of well-studied plasmid core proteins (Supplementary Section S11) from the categories of replication, partitioning, conjugative DNA transfer, exclusion, and type IV secretion system (T4SS), based on the list provided by Thomas et al. [52]. On the other hand, we retrieved high-quality proteins labeled as "plasmid" (Supplementary Section S12) from the Swiss-Prot database by utilizing 4 crucial plasmid accessory functions as keywords: AMR, resistance to heavy metals, new metabolic process, and virulence factors. In this section, we validated the effectiveness of PlasGO in capturing the underlying features for classifying plasmid-specific functions by selecting 4 GO terms from the 2 curated lists for each GO category.

For each selected GO term, we categorized the proteins associated with it as positive samples, while considering the remaining proteins not associated with it as negative samples. An inspection of our curated dataset revealed an imbalanced distribution of data across all 12 selected GO terms, with the majority of proteins classified as negative samples. To enhance clarity in the visualization, an equal number of negative samples were randomly sampled to match the number of positive samples for each GO term. We employed t-SNE (t-distributed stochastic neighbor embedding) [53] for 2D visualization of both the original embedding from ProtTrans and the embeddings generated by PlasGO's BERT module. The comparison of embeddings for the 4 MF binary classifications is presented in Fig. 7, while the comparisons for BP and CC are shown in Supplementary Section S13. To quantitatively evaluate the separation of embeddings between the 2 classes, we consider the positive proteins and negative proteins as separate clusters. Then, we calculate the silhouette score (ranging from −1 to 1) based on these 2 clusters. We can observe that the PlasGO embeddings exhibit a clearer separation of protein functions compared to the ambiguous ProtTrans embeddings. This suggests that PlasGO effectively captures the latent features associated with plasmid-specific functions.

Identification of elusive GO term labels

Predicting all GO terms accurately poses a significant challenge due to its complex multilabel classification. To make our tool more practically useful, we prefer to sacrifice prediction resolution for higher precision. Based on the empirical experiments, we identified a few labels that are extremely difficult to predict. Specifically, if the term-centric AUPR score on the validation set of a label falls below 0.3, then this label is considered an elusive label. As a result, a total of 16 out of 172 MF labels, 21 out of 174

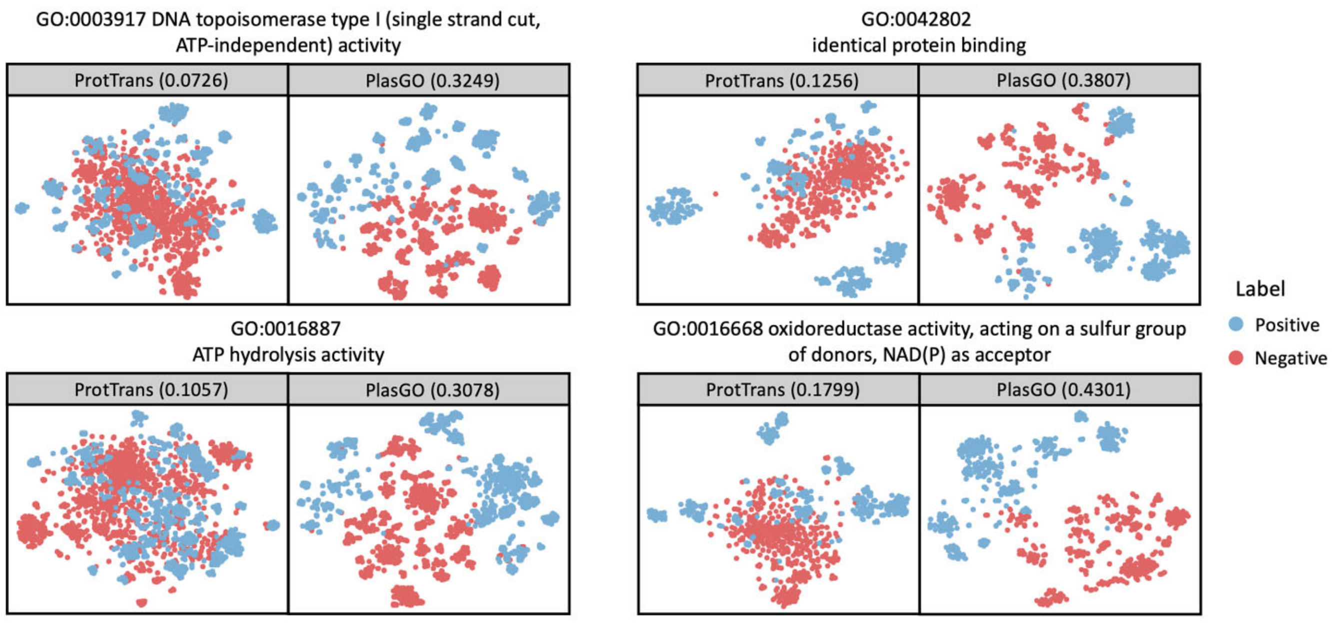


Figure 7: The visualization of protein embeddings generated by ProtTrans and PlasGO for 4 GO terms. For each GO term, the proteins associated with it and not associated with it are defined as positive and negative samples, respectively. Blue dots: positive samples; red dots: negative samples. The x-axis and y-axis represent the 2 dimensions of the embeddings reduced using t-SNE. Additionally, the silhouette score is displayed within brackets on each box.

BP labels, and 3 out of 31 CC labels have been identified as elusive labels. The detailed list of these elusive labels, along with their corresponding AUPR scores on the test set obtained from the top 4 tools (PlasGO, PFresGO, TM-Vec, and DeepGOPlus), is provided in Supplementary Section S14. It is notable that the majority of the elusive labels, with the exception of 5 of them (where PlasGO still outperforms the other tools), consistently achieve AUPR scores below 0.3 on the test set across all the top 4 tools. In summary, this suggests that current data cannot support reliable prediction of these labels.

As depicted in Supplementary Figure S8, the majority of elusive labels are rare classes. Thus, the presence of elusive labels can primarily be attributed to the scarcity of training samples, potentially hindering PlasGO's performance in 2 ways. First, the limited diversity within the training data for these elusive labels poses challenges for the model to effectively capture their distinctive features, thus impeding its ability to generalize to unseen proteins. Second, the imbalance between positive (proteins with the GO term) and negative samples (proteins without the GO term) associated with elusive labels may result in the model overfitting on negative samples and underfitting on positive samples. Consequently, the model tends to predict elusive labels in a "conservative" fashion, often refraining from positive predictions unless entirely confident. Moreover, due to our dataset-splitting strategy, the dissimilarity in protein sequences between the training set and test set is considerable (Supplementary Fig. S9). As a result, limited features can be used for predicting the novel functions of proteins in the test set. As shown in Supplementary Table S7, we filtered the results of the PlasGO module ablation study (Table 2) to specifically consider the elusive labels. The poor performance of the DNN classifier suggests that even the powerful PLM failed to capture the features necessary for predicting the elusive labels. Finally, the identified elusive labels have no overlap with the important plasmid-specific GO terms associated with both plasmid core proteins and accessory proteins, as detailed in Supplementary Sections S11 and S12. Hence, the functions represented by these elusive labels may not adhere to a distinct modularization pattern on plasmids, and as a result, PlasGO was unable to improve the prediction of these elusive labels by leveraging contextual information (Supplementary Table S7).

To address this issue, we have implemented 2 strategies. First, we introduced 2 usage modes for the PlasGO tool: the alignment-based mode and the learning-based mode. In the alignment-based mode, users can leverage Diamond [45] to conduct searches against our curated high-confidence protein database using a strict E-value cutoff (e.g., 1e-5). Notably, within our curated database, while we have filtered out the "elusive" PlasGO predictions, the original high-quality RefSeq annotations for the elusive labels are reserved. As a result, the alignment offers users precise annotations for these elusive labels when their query proteins exhibit homology to the corresponding target proteins. Second, the majority of the elusive labels have their ancestor terms reversed, as shown in the example DAG structure in Supplementary Fig. S10. As shown in Supplementary Fig. S11, the nearest ancestor terms of most elusive labels maintain good performance in terms of the AUPR metric. Thus, PlasGO can effectively predict the functions of elusive labels by prioritizing a broader representation of gene product attributes, albeit sacrificing some resolution.

Looking ahead, as high-quality GO annotations continue to expand rapidly, we anticipate that the PlasGO model will greatly benefit from the augmented training samples. This enhancement is expected to boost the performance of elusive labels, ultimately transforming them from "elusive" to labels that can be predicted with confidence.

Labels of different frequencies and confidence scores

We first conducted a more detailed evaluation of the performance comparisons on the RefSeq test set among the top 4 tools. Specifically, we focused on a few representative labels, namely, the first 10 and last 20 most frequent MF labels in the training set. In general, labels that occur more frequently tend to exhibit better per-

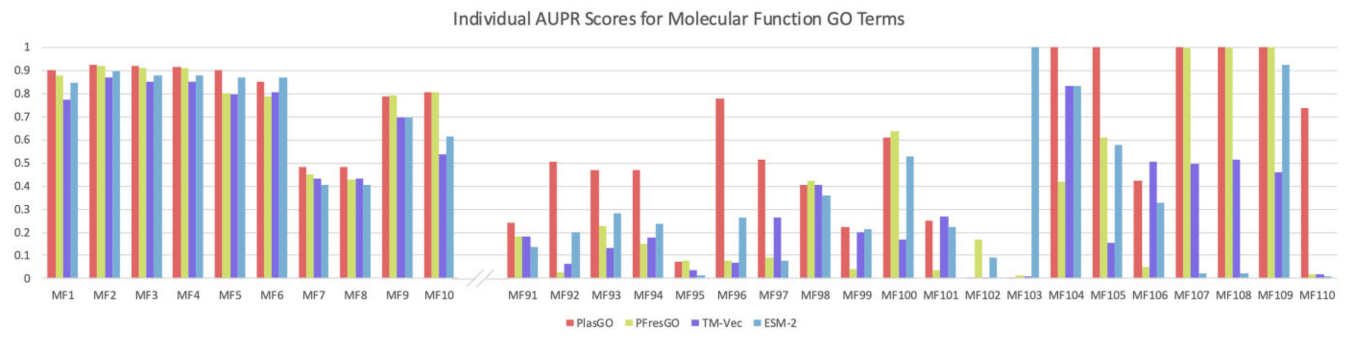


Figure 8: The AUPR comparisons of the top 4 tools on the first 10 and last 20 MF labels sorted by occurrence frequency in the training set. The performance of all tools fluctuates significantly on low-frequency labels.

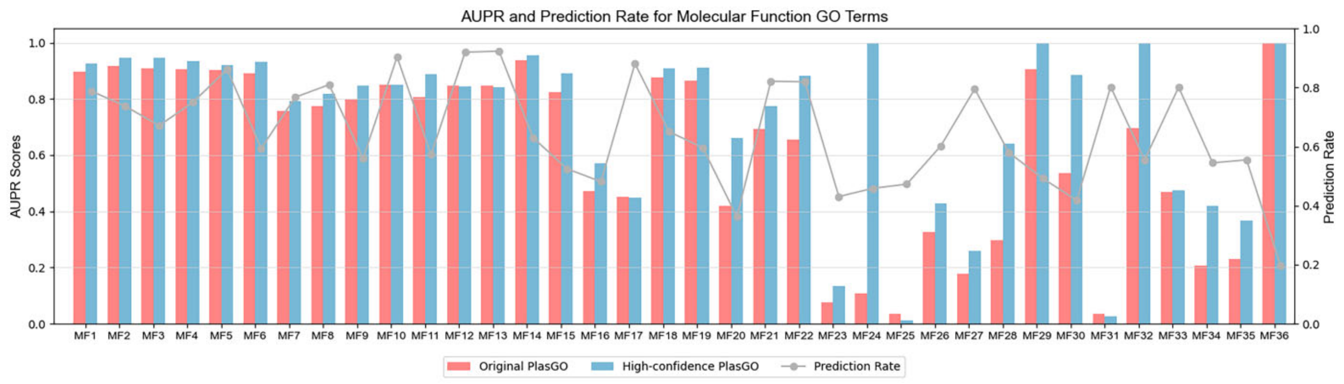


Figure 9: The AUPR comparisons on the MF category between the original PlasGO and the high-confidence mode of PlasGO. In the high-confidence mode, PlasGO filters out some predictions with low learned confidence scores, resulting in higher AUPR scores but lower prediction rates. The gray line shows the prediction rate for the "high-confidence PlasGO" mode. The prediction rates for the original PlasGO are all equal to 1 and thus are not shown in this figure. The corresponding comparison results on the BP and CC categories are shown in Supplementary Section S15.

formance. This can be attributed to the abundance of training samples, which allows the models to acquire richer features for effectively distinguishing these labels. This pattern aligns with the results presented in Fig. 8, which illustrates that the performance of the top 10 labels is superior and more stable compared to the performance of the last 20 labels. Despite obtaining low AUPR scores for several low-frequency labels, PlasGO consistently outperforms the other 3 tools across the majority of labels. Notably, several low-frequency labels attained a perfect AUPR score of 1, which is unusual. This anomaly is attributed to their associated limited number of test proteins. For example, the top 10 MF labels (ranging from *MF1* to *MF10*) average 4,544 associated test proteins out of a total of 17,369 test proteins. Conversely, the 5 MF labels at the end of Fig. 8, each achieving an AUPR score of 1, have an average of only 18 test proteins. As a result, low-frequency labels are more susceptible to random chance and tend to display significant fluctuations compared to high-frequency labels.

Next, we assessed the effectiveness of the learned confidence scores. As described in the Methods section, as the default setting, any prediction with a predicted probability below 0.425 and a confidence score lower than 0.95 will be excluded when calculating the AUPR for the high-confidence mode of PlasGO. To quantify this exclusion, we introduced a new metric called the "prediction rate," which is calculated by dividing the number of reserved testing proteins by the total number of testing proteins. Fig. 9 presents the performance comparison between the original PlasGO and its high-confidence mode for a subset of the MF labels, where the prediction rates are not equal to 1 in the high-confidence mode. We can observe that the high-confidence mode of PlasGO attains higher AUPR scores for the majority of the displayed labels by sacrificing a certain degree of prediction rate.

Application: automatic GO prediction for unannotated plasmid-encoded proteins in RefSeq

One primary contribution of PlasGO is its utilization of trained models to predict high-confidence GO term annotations for unannotated plasmid-encoded proteins that are not included in the training, validation, and test sets. As shown in Table 4, a large proportion of proteins curated from the RefSeq database (678,197) lack GO term annotations, with percentages of 74.39% for MF, 85.26% for BP, and 95.86% for CC. Hence, we employed PlasGO to predict nominal-format GO term annotations for these unannotated proteins. Consequently, we have successfully assigned high-confidence GO term annotations to most (all exceeding 95%) of the unannotated proteins. The distributions of the number of newly assigned GO terms for the 3 GO categories are presented in Supplementary Section S16. Lastly, the precision of the nominal-format GO term annotations was measured on the RefSeq test set, resulting in values of 0.8229, 0.7941, and 0.887 for the MF, BP, and CC categories, respectively. This confirms the high reliability of the newly added GO terms by PlasGO.

To conduct a more detailed analysis of the newly annotated proteins, we additionally provided the count of proteins in Table 4 that can be confidently classified into 1 of the 3 main core functions (replication, stability, and conjugation) using PlasGO. In

Table 4: The percentages of newly added high-confidence annotations by PlasGO across 3 GO categories. The last 3 columns display the count of proteins that have been newly predicted as having functions related to replication, stability, and conjugation, respectively, utilizing the significant GO term indicators.

# of all proteins	GO category	# of newly added annotations by PlasGO / # of unannotated proteins	Replication	Stability	Conjugation
	MF	488,696/504,531 (96.86%)			
678,197	BP	551,246/578,252 (95.33%)	136,303	38,836	22,630
	CC	650,012/650,116 (99.98%)			

order to achieve this classification, we manually identified the GO terms that exhibit strong relevance as indicators for the 3 core functions. The comprehensive list of these indicators can be found in Supplementary Section S18. Consequently, any protein predicted to have at least one of these GO term indicators will be classified into the corresponding core function. The results show that PlasGO successfully predicts a significant number of unannotated proteins as plasmid core proteins. Furthermore, we choose to show whether our annotated GO terms can reveal the functions related to plasmid replication. For this purpose, we applied PlasGO to proteins collected by PlasmidFinder [54], which includes 481 replicon sequences obtained from the PCR-based replicon typing (PBRT) scheme. The findings are elaborated in Supplementary Section S17. Finally, the predicted GO terms for plasmid-encoded proteins were saved in a database alongside PlasGO. Users can first attempt sequence alignment against the database with high identity and coverage cutoffs to directly obtain annotated GO terms. If there is no significant alignment, users can then apply the PlasGO models for GO term prediction.

Case study: annotations for 2 well-studied plasmids

We then evaluated PlasGO's performance in protein function prediction for the 2 well-studied conjugative plasmids associated with AMR, namely, pSK41 from *Staphylococcus aureus* [55] and pOLA52 from *Escherichia coli* [15]. The 2 complete plasmids, along with their corresponding encoded proteins, were downloaded from the NCBI database using the sequence IDs *NC_005024* and *NC_010378*, respectively. For both plasmids, over half of the encoded proteins possess informative "gene product" annotations, excluding those labeled as "hypothetical protein" and "domain-containing protein." However, only a very small portion of these proteins (4 out of 76) have GO annotations available. Therefore, the "gene product" annotations can offer a potential scope of the GO term ground truth, which aids in evaluating the predictive accuracy of PlasGO for the unannotated proteins. The detailed information on the proteins, including their protein IDs and "gene product" annotations, can be found in Supplementary Section S19, presented in the order corresponding to their encoding in the 2 plasmids. As shown in Fig. 10, to obtain the potential ground truth scope, we first retrieved the relevant proteins from the Swiss-Prot database by using the "gene product" annotation as the keyword phrase (e.g., "MobA/MobL family protein"). Following that, the collected GO terms associated with these retrieved proteins were considered as the potential ground-truth set for the corresponding protein. We can evaluate the predictive accuracy of PlasGO for each protein by calculating the Jaccard index between the potential ground-truth set and the GO terms predicted by PlasGO.

Figure 11 illustrates the number comparison between the predicted GO terms (merged from the MF, BP, and CC categories)

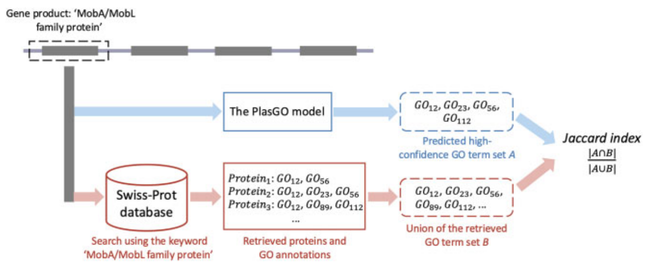

Gene product: 'MobA/MobL family protein'
The PlasGO model
$GO_{12}, GO_{23}, GO_{56}, GO_{112}$
Predicted high-confidence GO term set A
Jaccard index $\frac{|A \cap B|}{|A \cup B|}$
Swiss-Prot database
Search using the keyword 'MobA/MobL family protein'
$Protein_1$: GO_{12}, GO_{56}
$Protein_2$: $GO_{12}, GO_{23}, GO_{56}$
$Protein_3$: $GO_{12}, GO_{89}, GO_{112}$
...
Retrieved proteins and GO annotations
$GO_{12}, GO_{23}, GO_{56}, GO_{89}, GO_{112}, \dots$
Union of the retrieved GO term set B

Figure 10: The pipeline of conducting the case study experiment. Only the proteins annotated with informative "gene product" annotations were considered in this experiment. In this figure, we present an example evaluation of a protein annotated as "MobA/MobL family protein." The blue path illustrates the prediction process of PlasGO, wherein a set of high-confidence GO terms *A* is generated by our algorithm. The red path depicts the process of obtaining the potential scope of the GO term ground truth. Specifically, we conducted a search in the Swiss-Prot database using the phrase "MobA/MobL family protein" as the keyword. Subsequently, the proteins that matched the keyword, along with their corresponding GO annotations, were retrieved. Finally, we defined the union of the retrieved GO terms as the potential ground-truth set *B*. The predictive accuracy of PlasGO is evaluated by calculating the Jaccard index between the predicted high-confidence GO term set *A* and the potential ground-truth set *B*.

by PlasGO and the original GO terms available in the RefSeq database, along with the corresponding Jaccard index. PlasGO successfully added predicted GO terms to all the proteins and demonstrated a promising predictive accuracy for the majority of them. For enhanced clarity, the comparison of GO annotations for proteins encoded on plasmid pSK41 between the raw RefSeq database and the predictions generated by PlasGO is illustrated in Supplementary Figure S15. It is noteworthy that the increase in the number of GO terms for the 4 originally annotated proteins can be attributed to PlasGO assigning GO terms to previously unrepresented categories. For example, the 31st protein (*WP_012291478*) in plasmid pOLA52 had only 9 GO term labels in the MF category initially. PlasGO predicts an additional 4 BP terms and 4 CC terms for this protein, resulting in a total of 17 annotated GO terms.

Discussion

In this study, we presented a tool named PlasGO aiming to provide GO term-based functional annotation for largely uncharacterized plasmid-encoded proteins. Due to segment transfer facilitated by recombination events or the movement of MGEs, plasmids frequently demonstrate a modularization pattern, wherein proteins with similar functions tend to be positioned in close proximity to each other. To leverage this characteristic, we represented each plasmid as a sentence and functionally related

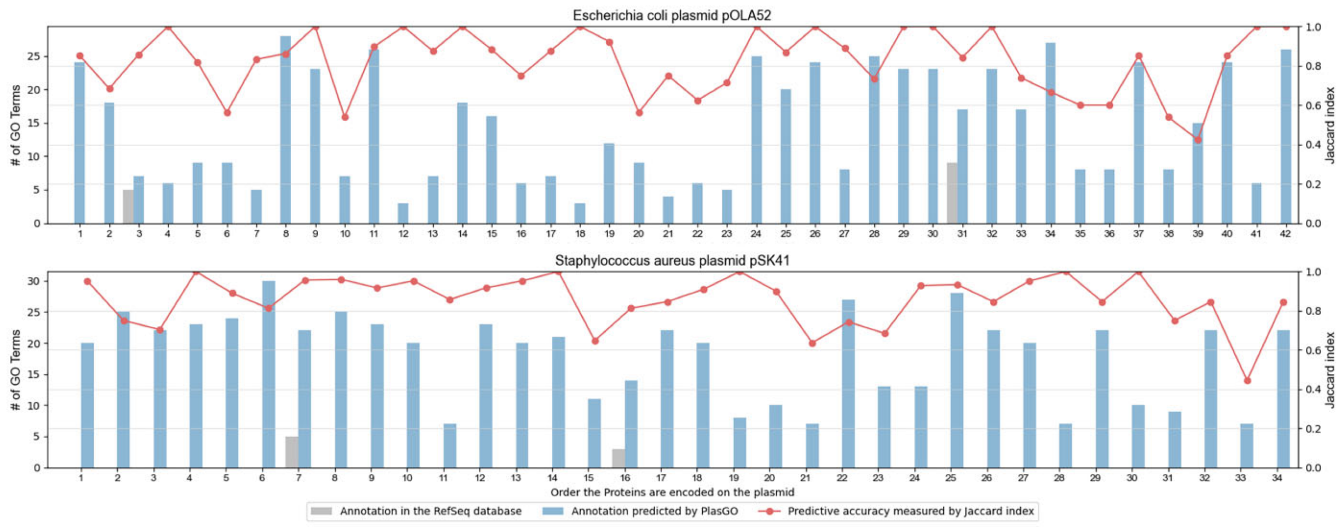


Figure 11: The comparisons between the number of GO terms predicted by PlasGO for the proteins encoded in the 2 well-studied plasmids and the number of original GO terms available in the NCBI RefSeq database. The indices on the x-axis represent the order in which proteins are encoded in the plasmids. The red lines in the figure represent the Jaccard index calculated between the high-confidence GO term set predicted by PlasGO and the potential ground-truth set retrieved from the Swiss-Prot database.

segments as phrases, both composed of multiple proteins. We then formulated the GO term prediction for plasmid-encoded proteins as a multilabel token classification task, utilizing the BERT model. PlasGO is specifically designed with a hierarchical architecture. It utilizes a foundation PLM to capture the local context within a protein sequence while employing a BERT model to capture the global context across different proteins. Furthermore, a classifier module featuring a self-attention confidence weighting mechanism is incorporated to generate confidence scores during prediction, thereby ensuring more reliable results.

We rigorously evaluated PlasGO using a series of experiments and benchmarked its performance with other SOTA tools. Specifically, to mimic the real-world challenges in GO term prediction for novel proteins that cannot be characterized with homology search due to low sequence identity, we constructed a test set that lacks significant sequence alignments to the training set. In order to maintain fairness in the comparisons, we retrained models or classifiers for all the other tools by using the same training set as PlasGO. The results from all the experiments consistently demonstrate the superior performance of PlasGO compared with other SOTA tools. Besides, we conducted a careful ablation study to investigate the contribution of different components of PlasGO to performance improvement. These prove that the advantage of PlasGO is attributed to its algorithm design rather than the mere augmentation of training data. Furthermore, the benchmark results indicate that PlasGO has the potential to be extended beyond plasmid-specific applications and applied to general GO term prediction tasks. In the final phase, we utilized PlasGO to predict high-confidence GO terms for the complete set of available plasmid-encoded proteins, totaling 678,197 proteins. Remarkably, over 95% of previously unannotated proteins were successfully assigned new GO terms across all 3 GO categories. The predicted high-confidence GO terms for all plasmid-encoded proteins will be compiled into a database, which serves as a contribution to the community interested in plasmids.

Despite the notable improvement in function prediction, PlasGO does have 2 limitations. First, only 29.34% of the proteins in the current database possess GO term annotations. In the most extreme cases, only a single protein in a sentence may have training labels. This limitation restricts the model's ability to effectively learn plasmid patterns and hampers their generalization capabilities. Second, despite the high quality of the GO annotations, different proteins may vary in their level of annotation. In other words, some proteins may have annotations at a very detailed level, while others may be annotated to broader, higher-level GO terms. Consequently, the multilabel classification task in PlasGO is inevitably affected by the issue of missing labels, leading to the introduction of label noise. The relabeling method presents a potential approach to address both limitations. Specifically, after the model has undergone training for a specified number of epochs or has converged, labels with high predicted probabilities and confidence scores can be assigned as the ground truth for their corresponding proteins, regardless of whether these proteins were initially annotated or unannotated. Consequently, as training progresses, an increasing number of proteins will no longer be masked and will actively contribute to the loss computation. On the other hand, the model will be fully utilized to rectify the missing labels within the dataset, enhancing the accuracy of the predictions.

PlasGO is implemented as an end-to-end and user-friendly tool. It can accept both complete plasmids and plasmid-borne contigs as inputs, generating high-confidence GO annotations for the proteins they encode. Users have the flexibility to provide meticulously curated plasmids from public databases like PLSDB [56] and IMG/PR [57]. Conversely, they can also initiate the analysis from diverse data sources, spanning isolates, metagenomes, metatranscriptomes, and single amplified genomes. Many tools, such as PLASMe [58] and MOB-recon [59], support the identification of plasmid-borne contigs from sequencing data. PlasGO can then integrate with gene recognition tools like Prodigal [60] to annotate protein functions for the input contigs. The predicted function annotations for plasmid-encoded proteins play a pivotal role in a range of downstream plasmid-related tasks. These tasks include plasmid mobility classification, plasmid replication mechanism prediction, and identifying accessory traits such as antibiotic resistance, virulence, and resistance to heavy metals. Furthermore, PlasGO can offer insights into more intricate aspects of plasmid research, encompassing host prediction and tracing the pathways of plasmid exchange among bacteria. A potential constraint of PlasGO in practical applications is the presence of elusive labels, which denote a few GO terms that are challenging to predict accurately. We have elaborated on the causes of these elusive labels, the current strategies implemented to address them, and potential future enhancements aimed at eliminating them with the increasing high-quality GO annotations in the "Identification of elusive GO term labels" section.

The architecture of PlasGO is flexible, enabling it to adapt to various functional annotation schemes for plasmid-encoded proteins. On one hand, instead of using GO terms, alternative training labels such as Enzyme Commission (EC) numbers [61], UniProtKB keywords [62], or even customized plasmid-specific protein function classes (e.g., replication and conjugation, though manual labeling efforts may be required) can be employed. On the other hand, PlasGO holds the potential to enhance the performance of the 3 principal plasmid typing schemes by predicting the classes of the associated proteins [5]. Specifically, Rep typing is based on replication initiation proteins, MOB typing relies on relaxases, and MPF typing is centered on T4SS proteins.

Availability of Source Code and Requirements

- Project name: PlasGO
- Project homepage: https://github.com/Orin-beep/PlasGO
- Operating system: Linux or Ubuntu
- Programming language: Python
- Other requirements: Python 3.x, NumPy 1.25, PyTorch>1.8.0, biopython, datasets, transformers, sentencepiece, accelerate
- License: MIT
- RRID: SCR_025983
- Bio.tools ID: PlasGO

Additional Files

Supplementary Fig. S1. The visualization of contextualized embeddings learned by PlasGO for the GO term "response to antibiotic."

Supplementary Fig. S2. The variations of average precision and recall, F1-score, $m(\theta)$ with different cutoff θ ranging from 0 to 1.

Supplementary Fig. S3. The matrix illustrates the all-against-all normalized McNemar test statistics between the ground truth, PlasGO, and the 7 benchmarked tools on the Molecular Function (MF) category. Each cell displays specific values, where a value close to 1 indicates a significant difference, while a value approaching 0 signifies the opposite.

Supplementary Fig. S4. The performance of PlasGO and the top 3 benchmarked tools on test sets derived from the 4 leave-one-genus-out experiment groups, evaluated using 2 metrics, Fmax (left) and AUPR (right), and assessed across the 3 GO categories. The 4 rows correspond to the comparison results of proteins within the genera *Escherichia*, *Klebsiella*, *Salmonella*, and *Enterococcus*, respectively.

Supplementary Fig. S5. The performance of PlasGO and the top 3 benchmarked tools averaged from the 5-fold cross-validation with the plasmid-based dataset split strategy. The results are evaluated using 2 metrics, (A) Fmax and (B) AUPR, and assessed across the 3 GO categories.

Supplementary Fig. S6. Embedding comparisons for the BP binary classifications.

Supplementary Fig. S7. Embedding comparisons for the CC binary classifications.

Supplementary Fig. S8. Sorted occurrence frequency of GO term labels in the training set across 3 GO categories. The x-axis represents the ranks of the GO term labels based on their frequency, while the y-axis represents the frequency in exponential format (base 10). The red bars indicate the elusive labels, while the blue bars represent the remaining labels. It can be observed that most of the elusive labels are rare classes.

Supplementary Fig. S9. The distribution of the distance between the training set and the test set for each elusive label. The distance distribution was measured by the minimum edit distance between each testing protein and the training set, normalized by dividing by the length of the longest sequence in the corresponding protein pair. The black, blue, and green boxes represent MF, BP, and CC labels, respectively. Furthermore, within each GO category, the elusive labels are sorted by their occurrence frequency in the training set. We can observe that all the minimum distances exceed 67.5, a low sequence similarity between the training set and test set for each elusive label [8].

Supplementary Fig. S10. The directed acyclic graph (DAG) structure, including the 3 CC elusive labels (shown as red boxes) and their ancestor terms (shown as blue boxes).

Supplementary Fig. S11. The performance comparisons between the elusive labels and their nearest ancestor terms for the 3 GO categories. Notably, the 3 MF elusive labels that exhibited poor performance on the validation set but good performance on the test set, achieving AUPR scores of 0.8165, 1.0, and 0.5, respectively, are not shown. Besides, a suffix "(top)" is added to 2 of the BP elusive labels, which indicates that they do not have reserved ancestor terms.

Supplementary Fig. S12. The AUPR comparisons on the BP category between the original PlasGO and the high-confidence mode of PlasGO.

Supplementary Fig. S13. The AUPR comparisons on the CC category between the original PlasGO and the high-confidence mode of PlasGO.

Supplementary Fig. S14. The top 10 predicted GO terms with the highest number of associated proteins. Each bar represents the ratio of annotated proteins out of the total 451 proteins for each respective GO term.

Supplementary Fig. S15. Comparison of GO annotations for proteins encoded on plasmid pSK41 between the raw RefSeq database (above) and predictions generated by PlasGO (below). The proteins are classified into 5 functional classes using the respective GO term indicators. We can observe that the raw RefSeq database contains GO annotations for only 2 proteins within the payload functional class. In contrast, PlasGO effectively assigned GO annotations to all proteins encoded on plasmid pSK41, with the exception of 2 proteins (highlighted by purple pentagon blocks) that could not be categorized into the 5 functional classes using the GO term indicators.

Supplementary Table S1. Performance comparison between PlasGO and the other top 3 tools on proteins larger than 1 Kbp.

Supplementary Table S2. Comprehensive breakdown of computational resources (maximum GPU memory usage and runtime) for each phase of PlasGO tested using a single NVIDIA RTX 3090 GPU.

Supplementary Table S3. Performance comparison for PlasGO using different training methods on the RefSeq test set. The last column indicates the round at which early stopping occurred during iterative fine-tuning due to performance decrease on the validation set.

Supplementary Table S4. Performance comparison between PlasGO and the classifier based on gLM's embeddings on the RefSeq test set.
Supplementary Table S5. Performance comparison between PlasGO trained upon ProtT5 and gLM on the RefSeq test set.
Supplementary Table S6. Detailed list of the elusive labels identified using the validation set. The third column represents the AUPR scores on the validation set for the elusive labels obtained from PlasGO, all of which are below 0.3. Furthermore, the fourth to seventh columns indicate the AUPR scores on the test set for the elusive labels obtained from PlasGO, PFresGO, TM-Vec, and DeepGOPlus (the top 4 tools), respectively. AUPR scores on the test set that exceed 0.3 are displayed in dark red.
Supplementary Table S7. The performance of different classification methods on the elusive labels evaluated using the RefSeq test set.

Ethical Considerations and Potential Commercial Implications

The primary dataset we utilized in this article is the publicly available NCBI RefSeq plasmid database [35]. The data included in this database are not associated with personal information or sensitive data. Therefore, there are no direct ethical considerations in the context of our work, and all the data we used comply with the terms and conditions set forth by NCBI for public use. Furthermore, the user-friendly PlasGO tool, along with the related data and code used in our experiments, is freely available on GitHub and Zenodo. As the article was submitted to *GigaScience*, an open-access journal, the tool is intended for broad, unrestricted use. Thus, we do not foresee any direct commercial implications arising from the use of PlasGO, as our focus remains on contributing to the public domain and supporting open science initiatives.

Abbreviations

AA: amino acid; AMR: antimicrobial resistance; AUPR: area under the precision–recall curve; BCE: binary cross-entropy; BP: Biological Process; CC: Cellular Component; CDS: coding sequence; CNN: convolutional neural network; DAG: directed acyclic graph; DL: deep learning; DNN: deep neural network; EC: Enzyme Commission; FC: fully connected; GAP: global average pooling; GO: Gene Ontology; HGT: horizontal gene transfer; MF: Molecular Function; MGE: mobile genetic element; MLM: masked language modeling; MOB: mobilization; MPF: mate–pair formation; NER: named entity recognition; NLP: natural language processing; PLM: protein language model; Rep: replicon; RR: rank regularization; SOTA: state-of-the-art; T4SS: type IV secretion system; TM: template modeling; t-SNE: t-distributed stochastic neighbor embedding; WBCE: logit-weighted binary cross-entropy.

Author Contributions

Yanni Sun (Methodology [Lead], Writing—review & editing [Lead]), Yongxin Ji (Data curation [Lead], Methodology [Lead], Software [Lead], Writing—original draft [Lead], Writing—review & editing [Equal]), Jiayu Shang (Methodology [Lead]), Jiaojiao Guan (Methodology [Supporting]), Wei Zou (Methodology [Supporting]), Herui Liao (Data curation [Supporting], Validation [Supporting]), Xubo Tang (Data curation [Supporting], Validation [Supporting]).

Funding

This work is supported by Research Grants Council (RGC), the University Grants Committee (UGC) of Hong Kong, General Research Fund (GRF) 11214924 (Y. Sun).

Data Availability

All codes and relevant data have been uploaded to the Zenodo repository [14005015] [63]. This encompasses all experimental data and codes utilized in the PlasGO paper, along with the scripts for preprocessing datasets, training models, and the user-friendly and end-to-end PlasGO tool implemented with Python. Snapshots of our code and other data further supporting this work are openly available in the *GigaScience* repository, GigaDB [64].

Competing Interests

The authors declare that they have no competing interests.

References

1. Smillie C, Garcillán-Barcia MP, Francia MV, et al. Mobility of plasmids. Microbiol Mol Biol Rev 2010;74(3):434–52. https://doi.org/10.1128/MMBR.00020-10.
2. Grohmann E, Muth G, Espinosa M. Conjugative plasmid transfer in gram-positive bacteria. Microbiol Mol Biol Rev 2003;67(2):277–301. https://doi.org/10.1128/MMBR.67.2.277-301.2003.
3. Rodríguez-Beltrán J, DelaFuente J, Leon-Sampedro R, et al. Beyond horizontal gene transfer: the role of plasmids in bacterial evolution. Nat Rev Microbiol 2021;19(6):347–59. https://doi.org/10.1038/s41579-020-00497-1.
4. Dewan I, Uecker H. A mathematician's guide to plasmids: an introduction to plasmid biology for modellers. Microbiology 2023;169(7):001362. https://doi.org/10.1099/mic.0.001362.
5. Shintani M, Sanchez ZK, Kimbara K. Genomics of microbial plasmids: classification and identification based on replication and transfer systems and host taxonomy. Front Microbiol 2015;6:242. https://doi.org/10.3389/fmicb.2015.00242.
6. Consortium GO. The Gene Ontology (GO) database and informatics resource. Nucleic Acids Res 2004;32(suppl 1):D258–61.
7. Nauman M, Ur Rehman H, Politano G, et al. Beyond homology transfer: deep learning for automated annotation of proteins. J Grid Comput 2019;17:225–37. https://doi.org/10.1007/s10723-018-9450-6.
8. Kulmanov M, Hoehndorf R. DeepGOPlus: improved protein function prediction from sequence. Bioinformatics 2020;36(2):422–29. https://doi.org/10.1093/bioinformatics/btz595.
9. Cao Y, Shen Y. TALE: Transformer-based protein function Annotation with joint sequence–Label Embedding. Bioinformatics 2021;37(18):2825–33. https://doi.org/10.1093/bioinformatics/btab198.
10. Pan T, Li C, Bi Y, et al. PFresGO: an attention mechanism-based deep-learning approach for protein annotation by integrating gene ontology inter-relationships. Bioinformatics 2023;39(3):btad094. https://doi.org/10.1093/bioinformatics/btad094.
11. Zheng J, Guan Z, Cao S, et al. Plasmids are vectors for redundant chromosomal genes in the Bacillus cereus group. BMC Genom 2015;16:1–10. https://doi.org/10.1186/1471-2164-16-1.
12. Pfeifer E, Rocha EP. Phage-plasmids promote recombination and emergence of phages and plasmids. Nat Commun 2024;15(1):1545. https://doi.org/10.1038/s41467-024-45757-3.

13. Finn RD, Attwood TK, Babbitt PC, et al. InterPro in 2017—beyond protein family and domain annotations. Nucleic Acids Res 2017;45(D1):D190–99. https://doi.org/10.1093/nar/gkw1107.
14. Hülter N, Ilhan J, Wein T, et al. An evolutionary perspective on plasmid lifestyle modes. Curr Opin Microbiol 2017;38:74–80. https://doi.org/10.1016/j.mib.2017.05.001.
15. Norman A, Hansen LH, She Q, et al. Nucleotide sequence of pOLA52: a conjugative IncX1 plasmid from Escherichia coli which enables biofilm formation and multidrug efflux. Plasmid 2008;60(1):59–74. https://doi.org/10.1016/j.plasmid.2008.03.003.
16. Le NQK. Leveraging transformers-based language models in proteome bioinformatics. Proteomics 2023;23(23–24):2300011. https://doi.org/10.1002/pmic.202300011.
17. Khan A, Lee B. DeepGene transformer: transformer for the gene expression-based classification of cancer subtypes. Expert Syst Appl 2023;226:120047. https://doi.org/10.1016/j.eswa.2023.120047.
18. Huang K, Xiao C, Glass LM, et al. MolTrans: molecular interaction transformer for drug–target interaction prediction. Bioinformatics 2021;37(6):830–36. https://doi.org/10.1093/bioinformatics/btaa880.
19. Tran TO, Le NQK. Sa-ttca: an svm-based approach for tumor t-cell antigen classification using features extracted from biological sequencing and natural language processing. Comput Biol Med 2024;174:108408. https://doi.org/10.1016/j.compbiomed.2024.108408.
20. Bepler T, Berger B. Learning the protein language: evolution, structure, and function. Cell Syst 2021;12(6):654–69. https://doi.org/10.1016/j.cels.2021.05.017.
21. Lin Z, Akin H, Rao R, et al. Language models of protein sequences at the scale of evolution enable accurate structure prediction. bioRxiv. 2022. https://doi.org/10.1101/2022.07.20.500902. Accessed 6 December 2024.
22. Devlin J, Chang MW, Lee K, et al. Bert: pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:181004805. 2018. https://doi.org/10.48550/arXiv.1810.04805. Accessed 6 December 2024.
23. Outeiral C, Deane CM. Codon language embeddings provide strong signals for use in protein engineering. Nat Mach Intel 2024;6(2):170–79. https://doi.org/10.1038/s42256-024-00791-0.
24. Elnaggar A, Heinzinger M, Dallago C, et al. Prottrans: toward understanding the language of life through self-supervised learning. IEEE Trans Pattern Anal Mach Intel 2021;44(10):7112–27. https://doi.org/10.1109/TPAMI.2021.3095381.
25. Brandes N, Ofer D, Peleg Y, et al. ProteinBERT: a universal deep-learning model of protein sequence and function. Bioinformatics 2022;38(8):2102–10. https://doi.org/10.1093/bioinformatics/btac020.
26. Li J, Sun A, Han J, et al. A survey on deep learning for named entity recognition. IEEE Trans Knowl Data Eng 2020;34(1):50–70. https://doi.org/10.1109/TKDE.2020.2981314.
27. Lin M, Chen Q, Yan S. Network in network. arXiv preprint arXiv:13124400. 2013. https://doi.org/10.48550/arXiv.1312.4400. Accessed 6 December 2024.
28. Raffel C, Shazeer N, Roberts A, et al. Exploring the limits of transfer learning with a unified text-to-text transformer. J Mach Learn Res 2020;21(140):1–67.
29. Jawahar G, Sagot B, Seddah D. What does BERT learn about the structure of language? In: Korhonen A, Traum D, Màrquez L, editors. ACL 2019-57th Annual Meeting of the Association for Computational Linguistics. Florence, Italy: Association for Computational Linguistics. 2019.
30. Zhao Y, Wang J, Chen J, et al. A literature review of gene function prediction by modeling gene ontology. Front Genet 2020;11:400. https://doi.org/10.3389/fgene.2020.00400.
31. Wang K, Peng X, Yang J, et al. Suppressing uncertainties for large-scale facial expression recognition. In: Boult T Medioni G Zabih R, editors. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. Seattle, WA, USA: IEEE. 2020:6897–906.
32. O'Leary NA, Wright MW, Brister JR, et al. Reference sequence (RefSeq) database at NCBI: current status, taxonomic expansion, and functional annotation. Nucleic Acids Res 2016;44(D1):D733–45. https://doi.org/10.1093/nar/gkv1189.
33. Jiang Y, Clark WT, Friedberg I, et al. The impact of incomplete knowledge on the evaluation of protein function prediction: a structured-output learning perspective. Bioinformatics 2014;30(17):i609–16. https://doi.org/10.1093/bioinformatics/btu472.
34. Zhou N, Jiang Y, Bergquist TR, et al. The CAFA challenge reports improved protein function prediction and new functional annotations for hundreds of genes through experimental screens. Genome Biol 2019;20:1–23. https://doi.org/10.1186/s13059-018-1612-0.
35. Reference sequence (RefSeq) plasmid database at NCBI. Bethesda (MD): National Library of Medicine (US), National Center for Biotechnology Information; 2004. https://ftp.ncbi.nlm.nih.gov/genomes/refseq/plasmid/. Accessed 25 November 2024.
36. Dicenzo GC, Finan TM. The divided bacterial genome: structure, function, and evolution. Microbiol Mol Biol Rev 2017;81(3):e00019–17. https://doi.org/10.1128/MMBR.00019-17.
37. Meier J, Rao R, Verkuil R, et al. Language models enable zero-shot prediction of the effects of mutations on protein function. Adv Neur Inf Proc Syst 2021;34:29287–303.
38. Pesquita C, Faria D, Falcao AO, et al. Semantic similarity in biomedical ontologies. PLoS Comput Biol 2009;5(7):e1000443. https://doi.org/10.1371/journal.pcbi.1000443.
39. Supek F, Bošnjak M, Škunca N, et al. REVIGO summarizes and visualizes long lists of gene ontology terms. PLoS One 2011;6(7):e21800. https://doi.org/10.1371/journal.pone.0021800.
40. Steinegger M, Söding J. Clustering huge protein sequence sets in linear time. Nat Commun 2018;9(1):2542. https://doi.org/10.1038/s41467-018-04964-5.
41. Suzek BE, Wang Y, Huang H, et al. UniRef clusters: a comprehensive and scalable alternative for improving sequence similarity searches. Bioinformatics 2015;31(6):926–32. https://doi.org/10.1093/bioinformatics/btu739.
42. Altschul SF, Madden TL, Schäffer AA, et al. Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. Nucleic Acids Res. 1997;25(17):3389–402. https://doi.org/10.1093/nar/25.17.3389.
43. Li S, Moayedpour S, Li R, et al. CodonBERT large language model for mRNA vaccines. Genome Res 2024;34(7):1027–35. https://doi.org/10.1101/gr.278870.123.
44. Hamamsy T, Morton JT, Blackwell R, et al. Protein remote homology detection and structural alignment using deep learning. Nat Biotechnol 2023;42(6):1–11.
45. Buchfink B, Xie C, Huson DH. Fast and sensitive protein alignment using DIAMOND. Nat Methods 2015;12(1):59–60. https://doi.org/10.1038/nmeth.3176.
46. Hwang Y, Cornman AL, Kellogg EH, et al. Genomic language model predicts protein co-regulation and function. Nat Commun 2024;15(1):2880. https://doi.org/10.1038/s41467-024-46947-9.

47. Zhang Y, Skolnick J. TM-align: a protein structure alignment algorithm based on the TM-score. Nucleic Acids Res 2005;33(7):2302–309. https://doi.org/10.1093/nar/gki524.
48. Xu J, Zhang Y. How significant is a protein structure similarity with TM-score= 0.5? Bioinformatics 2010;26(7):889–95. https://doi.org/10.1093/bioinformatics/btq066.
49. Yang Z, Dai Z, Yang Y, et al. Xlnet: generalized autoregressive pretraining for language understanding. Adv Neur Inf Proc Syst 2019;32:5754–64.
50. Lan Z, Chen M, Goodman S, et al. Albert: A lite bert for self-supervised learning of language representations. arXiv preprint arXiv:190911942. 2019. https://doi.org/10.48550/arXiv.1909.11942. Accessed 6 December.
51. Lichtarge O, Bourne HR, Cohen FE. An evolutionary trace method defines binding surfaces common to protein families. J Mol Biol 1996;257(2):342–58. https://doi.org/10.1006/jmbi.1996.0167.
52. Thomas CM, Thomson NR, Cerdeño-Tárraga AM, et al. Annotation of plasmid genes. Plasmid 2017;91:61–67. https://doi.org/10.1016/j.plasmid.2017.03.006.
53. Van der Maaten L, Hinton G. Visualizing data using t-SNE. J Mach Learn Res 2008;9(11):2579–605.
54. Carattoli A, Hasman H. PlasmidFinder and in silico pMLST: identification and typing of plasmid replicons in whole-genome sequencing (WGS). Methods Mol Biol 2020:2075;285–94. https://doi.org/10.1007/978-1-4939-9877-7_20.
55. Liu MA, Kwong SM, Jensen SO, et al. Biology of the staphylococcal conjugative multiresistance plasmid pSK41. Plasmid 2013;70(1):42–51. https://doi.org/10.1016/j.plasmid.2013.02.001.
56. Schmartz GP, Hartung A, Hirsch P, et al. PLSDB: advancing a comprehensive database of bacterial plasmids. Nucleic Acids Res 2022;50(D1):D273–78. https://doi.org/10.1093/nar/gkab1111.
57. Camargo AP, Call L, Roux S, et al. IMG/PR: a database of plasmids from genomes and metagenomes with rich annotations and metadata. Nucleic Acids Res 2024;52(D1):D164–73. https://doi.org/10.1093/nar/gkad964.
58. Tang X, Shang J, Ji Y, et al. PLASMe: a tool to identify PLASMid contigs from short-read assemblies using transformer. Nucleic Acids Res 2023;51(15):e83. https://doi.org/10.1093/nar/gkad578.
59. Robertson J, Nash JH. MOB-suite: software tools for clustering, reconstruction and typing of plasmids from draft assemblies. Microbial Genom 2018;4(8):e000206.
60. Hyatt D, Chen GL, LoCascio PF, et al. Prodigal: prokaryotic gene recognition and translation initiation site identification. BMC Bioinform 2010;11:1–11. https://doi.org/10.1186/1471-2105-11-119.
61. Bairoch A. The ENZYME database in 2000. Nucleic Acids Res 2000;28(1):304–5. https://doi.org/10.1093/nar/28.1.304.
62. Boutet E, Lieberherr D, Tognolli M, et al. UniProtKB/Swiss-Prot, the manually annotated section of the UniProt KnowledgeBase: how to use the entry view. Methods Mol Biol 2016;1374:23–54. https://doi.org/10.1007/978-1-4939-3167-5_2.
63. Ji Y. PlasGO_dataset. Zenodo. 2024. https://doi.org/10.5281/zenodo.14005015. Accessed 6 December.
64. Ji Y, Shang J, Guan J, et al.. Supporting data for “PlasGO: Enhancing GO-Based Function Prediction for Plasmid-Encoded Proteins Based on Genetic Structure.” GigaScience Database. 2024. http://dx.doi.org/10.5524/102621.

Received: July 4, 2024. **Revised:** October 29, 2024. **Accepted:** November 27, 2024