Transfer learning in proteins: evaluating novel protein learned representations for bioinformatics tasks

E. Fenoy, A. A. Edera, G. Stegmayer
Research Institute for Signals, Systems and Computational Intelligence
sinc(*i*) (CONICET-UNL), Ciudad Universitaria, Santa Fe, Argentina.

October 26, 2022

Abstract

Motivation: A representation method is an algorithm that calculates numerical feature vectors for samples in a dataset. Such vectors, also known as embeddings, define a relatively low-dimensional space able to efficiently encode high-dimensional data. Very recently, many types of learned data representations based on machine learning have appeared and are being applied to several tasks in bioinformatics. In particular, protein representation learning methods integrate different types of protein information (sequence, domains, etc.), in supervised or unsupervised learning approaches, and provide embeddings of protein sequences that can be used for downstream tasks. One task that is of special interest is the automatic function prediction of the huge number of novel proteins that are being discovered nowadays and are still totally uncharacterized. However, despite its importance, up to date there is not a fair benchmark study of the predictive performance of existing proposals on the same large set of proteins and for very concrete and common bioinformatics tasks. Therefore, this lack of benchmark studies prevent the community from using adequate predictive methods for accelerating the functional characterization of proteins.
Results: In this study, we performed a detailed comparison of protein sequence representation learning methods, explaining each approach and comparing them with an experimental benchmark on several bioinformatics tasks: (i) determining protein sequence similarity in the embedding space; (ii) inferring protein domains; and (iii) predicting ontology-based protein functions. We examine the advantages and disadvantages of each representation approach over the benchmark results. We hope the results and the discussion of this study can help the community to select the most adequate machine learning-based technique for protein representation according to the bioinformatics task at hand.
Availability: Full source code and data are available at:
https://github.com/sinc-lab/Comparison-of-Protein-learning

Contact: efenoy@sinc.unl.edu.ar

1 Introduction

Automatic functional annotation of proteins can be considered critical nowadays due to the rhythm of experimental data production [9]. For example, as of December 2021, there are around 220,000,000 protein entries in the UniProtKB; however, only 565,928 (less than 1%) of them have been reviewed and manually annotated by expert curators. This is a huge breach between sequencing and annotation capabilities.

This gap exists because of the high speed of experimental data obtention and, on the opposite, the very low and time-consuming manual curation of results. To aid experimental and curation-based annotation, automated in silico approaches can be used. Thus, many computational approaches have been proposed very recently [11, 39, 45, 54] to predict protein activities, properties, interactions, structure and functions [3, 22, 23, 41].

Automatic function prediction (AFP) is the algorithmic assignment of functional annotations —usually Gene Ontology (GO) terms— to proteins of unknown function from those whose function has already been determined experimentally [32]. Gene Ontology (GO) [8] is a hierarchical network of interconnected concepts that has become the standard vocabulary for describing protein function. The GO has more than 40,000 biological concepts over three sub-ontologies: Molecular Function (MF), Biological Process (BP) and Cellular Component (CC). The most comprehensive benchmark for AFP is the Critical Assessment of Functional Annotation (CAFA) challenge [56], in which participants predict GO-terms for target proteins. In the first two editions of the said challenge (CAFA1 (2010-2011) and CAFA2 (2013-2014)), there was a significant improvement between the participating methods, but in CAFA3 (2016-2017) this trend was interrupted, except for GOLabeler [54], which outperformed its competitors by using information (features) extracted from the sequence instead of the protein sequence alone. This was a critical improvement since, in many cases and especially in novel or recently sequenced species, the only available information is the sequence. That is why sequence-based methods for protein annotation prediction are highly used. Therefore, in spite of the recent breakthrough of alpha-fold's structure-based prediction of protein annotation [**?**], the amino acid sequence remains relevant and is, yet, the most widely-used data source for function prediction, followed by sequence-derived features such as domains and more recently, learned representations named embeddings [32]. This was a critical improvement since, in most cases, the only reliable information that can be found about a given protein is its sequence, implying the importance of sequence-based AFP. Nowadays, homology-based AFP tools such as BLAST are competitive, but they do not work well for proteins with less than 60% sequence identity to proteins with annotations. Together, these facts show that the problem of AFP is far from solved, it is still a big challenge and remains an open problem.

Table 1: Protein representation methods reviewed in this study.

Method	**Year**	**ML algorithm**	**Dimension**	**Repo**
CPCProt [31]	2020	MI	512	https://github.com/amyxlu/CPCProt
deepGOCNN [27]	2020	CNN	8,192	https://github.com/bio-ontology-research-group/deepgoplus
ESM-1b [46]	2020	Transformer-BERT	1,280	https://github.com/facebookresearch/esm
GP [53]	2018	Doc2vec	64	https://github.com/fhalab/embeddings_reproduction
PLUS-RNN [37]	2021	Bi-RNN	1,024	https://github.com/mswzeus/PLUS
ProtTrans/ProtBert [16]	2021	Transformer-BERT	1,024	https://github.com/agemagician/ProtTrans
ProtVec [3]	2015	Word2vec	300	https://github.com/ehsanasgari/Deep-Proteomics
rawMSA [38]	2019	CNN-LSTM	50	https://bitbucket.org/clami66/rawmsa
RBM [49]	2019	RBM	100	https://github.com/jertubiana/ProteinMotifRBM
SeqVec [19]	2019	LSTM-ELMO	1,024	https://github.com/Rostlab/SeqVec
TAPE [43]	2019	Transformer	2,048	https://github.com/songlab-cal/tape
UniRep [1]	2019	LSTM	1,900	https://github.com/churchlab/UniRep

In the last 6 years, several protein representation sequence-based models based on deep learning (DL) appeared, which given the raw sequence of a protein calculate a feature vector that is a unique representation of the protein, named embedding [36]. Then, a predictive model can efficiently learn the features of samples and perform the downstream prediction task by using these representations as input. This way, embedding models perform a process named transfer-learning of knowledge from one task to another [52]. Protein embedding has become a new and highly active area of research [12], being now actively and increasingly used by the community for the AFP task. For instance, DeepGOPlus predicts protein functions by first building protein sequence embeddings from raw protein sequences using multiple convolutional neural networks (CNN) [26]. Similarly, DEEPred predicts GO terms from several features automatically calculated from the protein sequence [45]. Meanwhile, goPredSim [30] makes annotation transfer based on similarity of protein-sequence embeddings obtained from DL models. Up to date, there are a lot of protein embedding methods available, which have all appeared after the CAFA3 and can be used for function prediction. However, there is no comparative study to systematically evaluate the performance of the myriad of available protein embedding methods with an experimental benchmark, and in the context of several and concrete bioinformatic tasks. For example, for the AFP problem, which is still unsolved [56], the correct selection of a protein representation method could boost the prediction methods available nowadays.

In this study, we provide a comprehensive review of the most recent protein embeddings available, with a benchmark analysis regarding the potential of these methods for the computational tasks of (i) determining protein sequence similarity in the embedding space; (ii) inferring protein domains; and (iii) predicting ontology-based protein functions.

2 Protein representation methods

In recent years, a myriad of protein embedding methods has appeared [32]. A comprehensive summary of protein embedding methods that were developed in the last 6 years in literature is presented in Table 1. The columns of Table 1 indicate each method name, year of publication, the ML algorithm it implements, the embedding dimension, whether the method allows a per-residue representation or not, and the corresponding public repository. The embedding methods listed in the table were included according to their availability, as open access tools or ready to use preconstructed feature vectors. A glossary of the terms used in the following descriptions of the methods can be found in Supplementary Material.

CPCProt [31]: this model uses supervised learning and it is based on maximizing the mutual information (MI) among proteins [48]. It proposes to use the information content of a certain number of protein sequence peptides (named patches) as a pre-training objective to capture motifs, structural elements, regions of unusual amino acid composition or parts of catalytic sites. Each patch is then passed through a multi-layer, convolutional neural network (CNN) encoder to obtain a latent representation.

Next, an autoregressive model produces a hidden state as well as a context latent representation based on the hidden state produced by the previous patch. In this way, the autoregressive model aggregates patch information, captured by the latent space of the autoencoder. The training task consists in finding an encoder able to map sequence fragments into meaningful representations, and an autoregressive model able to successfully summarize those representations of consecutive fragments, to construct a contextual embedding of a complete sequence. The CPCProt encoder consists of an embedding layer of 32 hidden dimensions, followed by 64 filters of length 4, 64 filters of length 6, and 512 filters of length 3. ReLU activation is added to the output of each layer, and normalization is added channel-wise. This model was pre-trained using protein domain sequences from the Pfam database [15], that is 32,207,059 amino acid sequences.

deepGOCNN [27]: this model is based on supervised learning and CNN [29]. It was the first deep learning model proposed for predicting protein functions from amino acid sequences of up to 2,000 residues. The resulting sequence is represented as a 21-to-2,000 one-hot matrix which is independently processed by 16 1-D convolutional layers with increasing filter lengths. Each layer has 512 equal-length filters able to detect specific sequence motifs of a particular size. The output of each filter is passed through a MaxPooling1D layer, transforming every filter into a single score representing the presence of a relevant motif in the input sequence. By stacking these scores, a feature vector of 8,192 components is built. DeepGOCNN was trained in a supervised way using protein sequences as inputs, and their corresponding GO terms annotations as target output, provided by CAFA [56] for more than 66,000 protein sequences from UniProtKB. The model has multiple 1D convolutional layers with different filter lengths where the smallest filter starts from length 8 and the following filters are increased by 8 units. The CNN layers do not use dropout because there is a MaxPooling layer with maximum pool size, thus every filter returns only a single value. This forces the CNN filters to learn a set of similar patterns (motifs) and if the filter finds the pattern in the sequence it returns a high value, which is pooled with the MaxPooling layer. For this comparative study, we used the feature vectors built by deepGOCNN as embeddings for our protein sequences.

ESM [46]: this model uses unsupervised learning and it is based on Transformers [50], which have emerged as a powerful general-purpose model architecture for representation learning and generative modeling, outperforming recurrent and convolutional architectures in natural language settings. ESM uses a deep Transformer BERT [14] that processes sequences of amino acids as input. BERT was originally designed for Natural Language Processing (NLP) based on unsupervised learning where context within a text is used to predict missing words. Its main hypothesis is that a word's semantics can be derived from the contexts in which it appears. ESM makes an analogy between words and sequences of amino acids. The model was trained using the masked language modeling objective, where each input sequence is corrupted by replacing a fraction of the amino acids with a special mask token, to predict the missing tokens from the corrupted sequence. ESM was trained on 220 million sequences in the UniProtKB database. In this comparison we have used ESM-1b because it was the best performing instance of the method according to its authors.

GP [53]: this model is supervised and based on doc2vec [28], which is in turn based on word2vec [35]. The latter approach learns embeddings or words with the skip-gram architecture, where the model uses the current word to predict its surrounding context words, and the continuous bag-of-words architecture, where the current word is predicted from its surrounding context words. The doc2vec model extends word2vec by learning embeddings for entire sentences, paragraphs, or documents. In the context of proteins, GP trains a doc2vec model to predict a central k-mer within a given protein based on local and global information. As local information, the flanking k-mers are used, whereas the embedding of the protein sequence is used as global information. GP was trained on 524,529 protein sequences obtained from UniProt, whose lengths range from 50 to 999 amino acids. After training, the embedding model was used to infer encodings for supervised Gaussian process (GP) regression models [44]. The embedding model is trained on randomized UniProt sequences (by shuffling the order of amino acids for each sequence, or by resampling sequences of the original lengths according to the overall observed amino acid frequency), instead of the original UniProt sequences, choosing its hyperparameters by using 20-fold cross-validation on the training sets. It was found by the authors that randomizing the UniProt sequences before unsupervised training had a regularizing effect, preventing the embedding model from overfitting to the set of proteins used for training. The rationale behind this is that the unsupervised embedding model is able to learn the frequency with which different amino acids occur in the same proteins. The noise and kernel GP hyperparameters were optimized by maximizing the marginal likelihood.

PLUS-RNN [37]: this model is based on supervised learning and its training involves 2 tasks: masked language modeling [14] that consists in predicting residues randomly masked in the input protein sequence, and same-family prediction [5] that involves predicting whether pairs of protein sequences belong to the same protein family. Given a pair of protein sequences as input, PLUS-RNN embeds each residue into a 21-dimensional vector, which is independently processed by a bidirectional recurrent neural network biRNN [17], of 3 layers. In each layer, there are 2 RNNs having long short-term memory (LSTM) [21] as activation units. LSTMs are a type of recurrent neural network capable of learning order dependence in sequence prediction problems. Recurrent networks have an internal state that can represent context information, and can keep information about past inputs for an amount of time that depends on its weights and on the input data. Thus, an input sequence can be transformed into an output sequence while taking into account contextual information [**?**]. One of the recurrent networks sequentially processes the sequence left-to-right to represent each residue as a new 1,024-dimensional vector given its context; the other RNN does the same, but right-to-left. The residue representations obtained by both networks are concatenated into a single representation, which encodes information from both sequence sides. PLUS-RNN was optimized with a dataset containing 14,670,860 protein sequences from 3,150 families publicly available in Pfam [15]. For this review, we averaged the vector representations per residue built by PLUS-RNN in the third layer of both recurrent networks.

ProtTrans [16]: this model was trained using several Transformer models [50], thus it is based on unsupervised learning. The authors trained two auto-regressive models (Transformer-XL and XLNet) and four auto-encoder models (BERT, Albert, Electra, T5) on data from UniRef50 and the BFD database [47] containing more than 2,000 million proteins and up to 393 billion amino acids. BERT (mentioned before) was the first model in NLP which tried to reconstruct corrupted tokens, and is considered the de-facto standard for transfer learning in NLP. Albert reduced BERT's complexity by hard parameter sharing between its attention layers. Electra tries to improve the sampling-efficiency of the pre-training task by training two networks, a generator and a discriminator. T5 consists of an encoder that projects a source language to an embedding space and a decoder that generates a translation to a target language based on the encoder's embedding. For ProtTrans, single amino acids were considered as input "words"; each protein sequence in a line represented the equivalent of "sentences"; an empty line was inserted between each protein sequence to indicate the "end of a document". The information learned by the protein learning models were the vector representations from the last hidden state of the Transformer. There are several ProtTrans instances available (ProtBert, ProtAlbert, ProtT5, ProtXLNet and ProtElectra). In this study, we used the instance ProtT5 trained with the BFD dataset and fine tuned on UniRef50 because it was the best performing method according to the authors [16].

ProtVec [3]: this model provides bio-vectors (BioVec) for biological sequences in general, protein-vectors (ProtVec) for proteins (amino-acid sequences) and gene-vectors (GeneVec) for gene sequences. Protein vectors were built with supervised neural networks that represent a protein sequence with n-gram modeling, where lists of shifted non-overlapping words are generated with window size 3. In the training process of word embedding in NLP, a large corpus of sentences should be fed into the training algorithm to ensure sufficient contexts are observed. Similarly, a large corpus is needed to train distributed representation of biological sequences. The next step is to break the sequences into subsequences (i.e. biological words), using n-grams directly in feature extraction. The embedding was trained through a Skip-gram neural network [35], which attempts to maximize the probability of observed word sequences (contexts). Such a constraint allows similar words to assume a similar representation in this space. Authors used Word2Vec [36] for word embedding since it was considered, at that time, as the best state-of-the-art method for training word vector representation. The procedure was applied on 546,790 manually annotated and reviewed sequences of the Swiss-Prot database, thus the training corpus had 546,790 $\times$ 3 = 1,640,370 sequences of 3-grams (a 3-gram was considered as a "biological" word consisting of 3 amino acids).

rawMSA [38]: this model builds an embedding that can be used to convert each residue character from a Multiple Sequence Alignment (MSA) into a floating-point vector of variable size. This is adaptively learned by a supervised deep neural network based on context. Unlike classical ML methods for the prediction of protein features, rawMSA does not compress the MSA into a profile but uses the raw aligned sequences as input and extracts useful features with a deep network. The input to the deep network is a flat FASTA alignment, where each letter is mapped to an integer ranging from 1 to 25 (standard residues and special cases). The first layer of rawMSA is a shallow two-layer neural network named an embedding layer. Then a 2D convolutional layer is stacked on top, followed by a max-pooling layer. Convolution is performed along each column in the MSA disallowing the information to spread across columns. The convolutional and pooling layers are followed by a stack of two bidirectional LSTM layers, where each module contains 350 hidden units. The final three layers are fully connected. All the convolutional layers have ReLU activations and the outputs are zero-padded to match the two first dimensions of the inputs. rawMSA was trained on 29,653 protein chains extracted from a 70% redundancy-reduced version of the Protein Data Bank (PDB) [51] in April 2017 with a minimum resolution of 3.0 Å and R-factor 1.0.

RBM [49]: here the authors, in contrast to any other method designed for this task, propose the usage of Restricted Boltzmann Machines (RBM) [20] for extracting the structural and functional features common to a protein family. RBM is a joint probabilistic model for sequences and representations. It is formally defined on a bipartite, two-layer graph. Protein sequences are displayed on the Visible layer, and representations on the Hidden layer. Each visible unit takes one out of 21 values (20 amino acids + 1 alignment gap). This model can capture structure-related, functional and phylogenetic features related to sub-families sharing evolutionary determinants. Training RBM required intensive Markov Chain sampling. This approach was tested with MSAs of 20 different protein families from Pfam. The training was performed by maximizing, through stochastic gradient with mini-batches of data, the difference between the log-probability of the sequences in the MSA and a regularization cost.

SeqVec [19]: this supervised model uses the bi-directional language model ELMo (Embeddings from Language Models) [42], which was originally designed for natural language tasks. It can be trained on unlabeled text-corpora to predict the most probable next word in a sentence. By learning a probability distribution for sentences, the model autonomously develops a notion for syntax and semantics of a language. The trained vector representations of a given word depend on its context, which has the advantage that two identical words can have different embeddings, depending on the words surrounding them. In SeqVec the ELMo concept was applied to model protein sequences by treating them as sentences and each amino acid as words. First, an input sequence is padded with special tokens indicating the start and the end of the sentence/protein sequence. Then character convolutions map each word/amino acid onto a fixed-length latent space without considering information from neighboring words. The output of the CharCNN-layer is used by a biLSTM that introduces context-specific information by processing the input sequentially. Finally, the 2nd LSTM-layer tries to predict the next word given all previous words in a sentence. The forward and backward pass are optimized independently in order to avoid information leakage. During inference, the hidden states of the forward and backward pass of each LSTM-layer are concatenated to a 1024-dimensional embedding vector summarizing information from the left and the right context. It was trained on UniRef50 (33 million sequences) using an architecture composed by a convolutional layer, followed by two bidirectional LSTM layers. ELMo was designed to reduce the risk of overfitting by sharing weights between the forward and the backward LSTMs and by using dropout. For our analyses, the vector representations per residue obtained by the two LSTM layers were concatenated and then averaged to obtain a single vector per protein.

TAPE [43]: in this case authors used Pfam [15] as the unsupervised pre-training corpus for TAPE. Training and testing sets were built using a random 95/5% split, respectively. Perplexity metrics were calculated for the held out families test set to measure out-of-distribution generalization to proteins less evolutionarily related to the training set. TAPE provides a 12-layer Transformer [50] with a hidden size of 512 units and 8 attention heads, having a 2,048 output size. The Transformer was trained with masked-token prediction, which models each data point by replacing the value of tokens at multiple positions with alternate tokens.

UniRep [1]: this supervised model uses a training sequence dataset consisting of 24,000,000 UniRef50 amino-acid sequence entries, which are filtered by a 50% similarity threshold from the UniProtKB. Authors trained a 1,900-hidden unit Multiplicative LSTM (mLSTM) with amino-acid character embeddings. These RNNs learn by going through a sequence of characters in order, trying to predict the next one based on the model's internal hidden state. During training, the model gradually revises the way it constructs its hidden state to maximize the accuracy of its predictions, resulting in a progressively better statistical summary, or representation, of the sequence. The model internal states were looked up in detail, finding that the amino-acid embeddings learned contained physicochemically meaningful clusters. It was also found that a single neuron positively correlated with alpha helix annotations, and negatively correlated with beta sheet annotations. The mLSTM architecture has two internal states that encode information about the sequence it is processing, the hidden state and the cell state. The output of the model is built with the concatenation of the final hidden state, final cell state, and average hidden state of the LSTM model, where the concatenation of the different states provides a protein representation vector with different levels of semantic information.

In summary, it can be stated that most models that approached protein representation as a supervised learning problem (such as CPCProt, DeepGOCNN and rawMSA) used similar CNN architectures, with small variants in the output layers (ReLU or MaxPooling). Very differently, PLUS-RNN, SeqVec and UniRep used recurrent neural networks and LSTMs; while GP and ProtVec proposed to use doc2vec and word2vec for extracting features from protein sequences. Nevertheless, it is worth noting that the architecture of SeqVec also includes a CNN, whose aim is to read the input protein sequences. In contrast, RBM uses a classic neural network with a very restricted architecture that was designed to reduce the computation cost of making predictions. Most of these supervised models were trained with UniProt and Pfam data. Regarding unsupervised models, it is very interesting to note that all of them were based on Transformers but with different architectures, and each of them trained in a different dataset. The best TAPE model had only 12 hidden layers and was trained on Pfam; while ESM and ProtTrans are those with more hidden layers (¿ 30) and both based on a pre-trained BERT.

3 Data, measures and experimental setup

3.1 Data

Proteins. We used a subset of the standard CAFA3 challenge dataset [56]. For this study, we selected the model species and highly annotated *Homo sapiens* proteins with lengths equal to or less than 700 amino acids, due to the restrictions imposed from some of the methods compared, resulting in a dataset of 9,479 molecules. In addition, to have comparative results from less annotated and non-model species, we used a subset of the protein sequences of *Rattus norvegicus* and *Mycobacterium tuberculosis H37Rv* from the CAFA3 dataset with a maximum length of 700 residues, obtaining 4,374 and 1,503 protein sequences, respectively.

Protein function annotations. Each protein had at least one GO term in any of the three sub-ontologies BP, MF and CC. The tags assigned to the given protein were propagated to the root of the corresponding sub-ontology using goatools [25] with the is-a and part-of relations. The Gene Ontology used was from the 2016-06-01 version, as it is contemporary to the CAFA3 dataset. In this way, proteins were enriched with annotations at several levels of the ontology tree, avoiding cases where a molecule is annotated only with a very specific term, isolating it from similar, but less studied, cases.

Domains. The corresponding information about protein domains for the 9,479 *Homo sapiens* proteins used in this study was obtained from the Pfam database [15] with the ProDy python package [55], making a total of 3,872 domains. For *Rattus norvegicus* and *Mycobacterium tuberculosis H37Rv* the number of domains is 2,474 and 1,199, respectively. It should be noted that the Pfam domains of each protein were compared with those annotated in UniProtKB, showing a 98% of agreement, thus the Pfam domains were used. Each protein can be composed of one or more domains, each playing a role in the protein global function. Some of these domains are tightly related to a given function, for example the catalytic domain of the kinase proteins; while others have a support role, such as DNA binding domains.

3.2 Performance measures

To evaluate how well protein embeddings cluster according to their annotated Pfam domains, we used a clustering index. To this aim, we considered Pfam domains as clusters of proteins and measured the degree of membership to clusters by using protein distances based on embeddings. Although a number of such indices have been proposed, the silhouette coefficient has shown superior performance for diverse problems. The silhouette coefficient is

$$\frac{1}{M}\sum_{i}^{M}\frac{b_i - a_i}{\max(a_i, b_i)}, \tag{1}$$

which is the average silhouette score over M proteins. This score is composed of two components: a_i is the mean distance between protein i and all the other proteins annotated with its same Pfam domain C_i

$$a_i = \frac{1}{|C_i| - 1}\sum_{j \in C_i, j \neq i} d(i, j), \tag{2}$$

where $|C_i|$ is the number of proteins annotated with domain C_i. Then b_i is the smallest mean distance of protein i to all the other proteins annotated with different domains

$$b_i = \min_{C_k \neq C_i} \frac{1}{|C_k|}\sum_{j \in C_k} d(i, j). \tag{3}$$

In both equations, d is the distance between protein i and protein j. The best value of the silhouette coefficient is $+1$ (well-separated clustering) and its worst value is -1 (invalid clustering). Values near 0 indicate overlapping clusters.

The predictive performance of the methods studied here are measured with standard recall or sensitivity (s^+) and F_1 evaluation metrics,

$$s^+ = \frac{TP}{TP + FN}, \tag{4}$$

$$F_1 = \frac{2TP}{2TP + FP + FN}, \tag{5}$$

where TP, TN, FP and FN are the number of true positives, true negatives, false positives and false negatives, respectively. In the context of this review, TP is the set of GO terms/Pfam domains associated with the protein to be measured; while FP and FN are the number of false positives and false negatives GO terms/PFam domains, respectively, predicted by a classification method. The recall measures how good a classification method is for recognizing the TPs of the task. It is also important to take into account the number of GO terms/Pfam domain families predicted but not related to the protein of interest, that is FP. Therefore, F_1 is used as a global comparative measure among methods. Friedman test and critical difference (CD) diagram with posthoc Nemenyi test [13] were used to assess the statistical significance of differences in the average F_1 achieved by each model when predicting over each protein in each test fold. In some experiments, the clustering coefficient of groups of proteins was also calculated to assess results.

In order to measure the similarity between protein embeddings, the cosine similarity was used, which is defined as

$$s_{cos} = \frac{< a, b >}{||a||||b||}, \tag{6}$$

where a and b are embedding vectors, the numerator is the dot product between the two vectors and $||\cdot||$ is the Euclidean norm. This similarity varies between -1 (total dissimilarity) and +1 (total similarity). For some experiments, we also used the cosine distance which is defined as $1 - s_{cos}$.

To measure the degree of correlation between sequence and cosine similarities obtained from pairs of protein sequences we calculated the Spearman correlation coefficient as

$$\rho = \frac{\operatorname{cov}(r(X), r(Y))}{\sigma_{r(X)}\sigma_{r(Y)}} \tag{7}$$

where X and Y are the BLAST for proteins (BLASTp) [2] similarities and cosine similarities, respectively; cov is the covariance between them, and σ are the standard deviations calculated separately for each set. Given a set as input, the function $r(\cdot)$ ranks the values within the set returning the positions that each value occupies when sorting them from the lowest to the highest. This formula measures how monotonically correlated the sets X and Y are, and varies between -1 (negative correlation) and +1 (positive correlation) with 0 indicating no correlation.

3.3 Experimental setup

In order to capture a higher order relationship between the embeddings and the GO terms/Pfam domains we used two machine learning (ML) methods widely used for classification tasks: k-nearest neighbor (kNN) [18], with a range between $k = 1$ and $k = 10$ and using the cosine similarity between embeddings for determining the neighborhoods; and a multilayer perceptron (MLP) neural network [7] with 3 hidden layers of size 1,024, 512 and 512, respectively, and an output layer of problem-specific size: 3,872 for Pfam domains; and 1,203 for CC, 3,159 for MF and 9,634 for BP sub-ontologies, respectively. The model architecture used for both Pfam and GO predictions is depicted in the Supplementary Figure S1. The MLP has ReLU [40] activation function for each layer, except the last layer which has a sigmoid activation to produce an output vector with normalized components. The MLP model was fed with the embedded proteins and used to predict the corresponding GO terms/PFam domains, encoded as binary vectors. To make predictions, the predicted vectors were transformed into binary vectors using a threshold of 0.3. This threshold was determined based on preliminary experiments aimed to study the impact of this value on predictions.

Model evaluation was performed as a typical 10-fold cross-validation experiment. The performance measurement is defined as the average s^+ and F_1 calculated from the test predictions for all folds. To address data redundancy, proteins with at least 80% sequence identity were forced to share the same partition. Additionally, a more restricted experiment was also performed with test data having less than 20%, 40% and 60% percentage pairwise sequence identity (PIDE) to the train set, which can be found in Supplementary Figure S2, where the overall behavior and differences among methods remain but the F_1 score is reduced since protein annotations that are representative in the test set are absent in the training set because of the strict partitioning. The model was trained for 1,000 epochs employing binary cross entropy [6] as the loss function; and the Adam optimizer [24] for weight optimization with mini-batches.

4 Results

4.1 Embedding space visualization

A protein embedding useful for downstream tasks is expected to represent crucial molecular features of proteins, such as their physicochemical properties as well as structural aspects. Protein embeddings encoding similar molecular features should, then, share a common region in the multidimensional embedding space. To have a very preliminary insight into this, a visual and just qualitative comparison was performed. The mathematical intuition behind the interpretation of the embeddings projections is that proteins with similar function should be close (near) in the projected embedding space, while functionally unrelated proteins should be separated and not overlapped. First, the protein embeddings were non-linearly reduced to 2 dimensions using UMAP [34], which has proved to be better for projecting the global structure of the embedding space [4]. After that, each molecule was painted with a different color, depending on its corresponding Pfam domain.

The 10 domains shown in Supplementary Table S1 were selected to highlight some interesting cases. For example, Immunoglobulin V-set (PF07686) and Immunoglobulin C1-set (PF07654) domains are both Ig-like domains involved in the immune system. Structural studies [10] have shown that both domains share a common structural core and also share a similar immunological function. Therefore, it would be expected for their corresponding embedding representations to be grouped in similar location or very close in the embedding space since both share similar sequence properties.

Figure 1 shows the UMAP visualization of all the 12 methods reviewed here only as a preliminary and broad qualitative analysis. As control, we used UMAP projections obtained from vectors representing each protein as its pairwise sequence similarities calculated with BLASTp. In each visualization, each Pfam family of Supplementary Table S1 is depicted with a different color. It can be seen that the families PF07686 (blue) and PF07654 (orange) are very close to each other, or even overlapped at least partially, in the embeddings of CPCProt, deepGOCNN, ESM, PLUS-RNN, ProtTrans, SeqVec, TAPE and UniRep. In contrast, this overlap is barely seen in the BLASTp projections, suggesting that classic sequence similarities among proteins are not discriminative enough. Very interestingly, in ProtTrans, SeqVec, ESM and TAPE, the blue and orange dots are partly overlapped and very separated from the rest of the protein families. However, in GP, ProtVec, RBM and rawMSA, those families are not close in the embedding space, in some cases, those are even located in opposite positions, far away.

Another interesting case to analyze is the kinase domains PF00069 (green) and PF07714 (red). Proteins annotated with PF00069 are serine/threonine kinases while PF07714 is assigned mainly to tyrosine kinases. Both groups share the same function. The red and green dots are together and overlapped, as expected, in all embeddings except in BLASTp, GP, RBM and rawMSA, where those are mostly distributed all over the space. It should be mentioned that in PLUS-RNN, SeqVec,

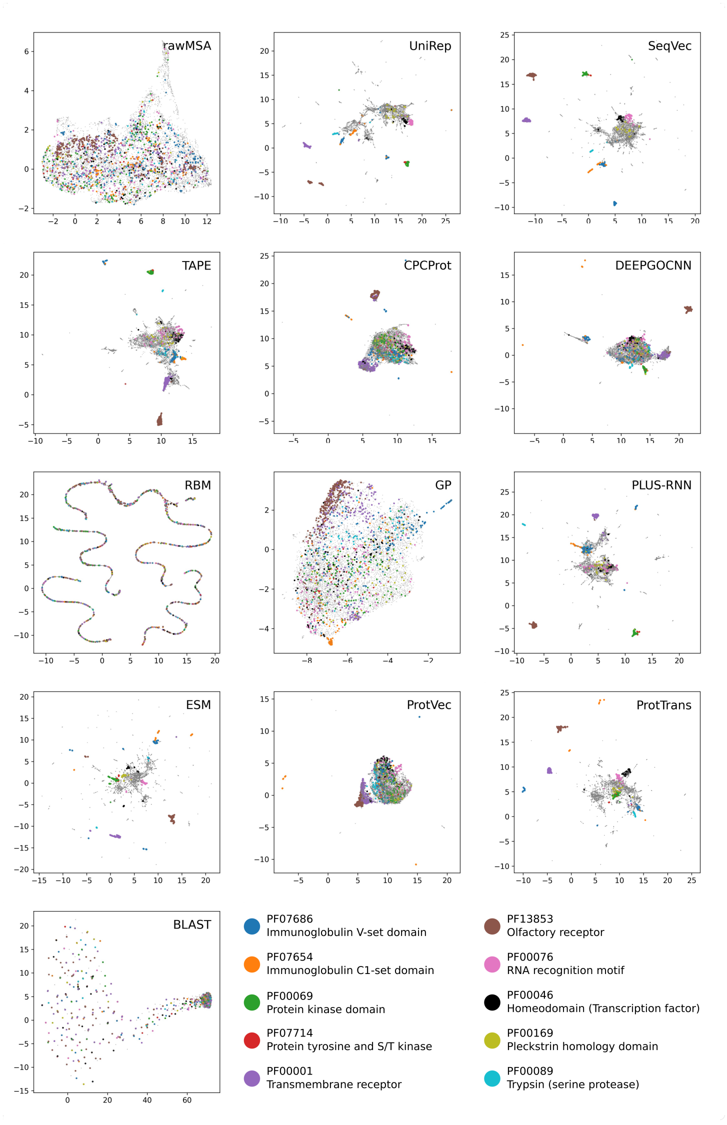

Figure 1: Visual comparison of the protein representation methods: visualization of the embedding space produced by each approach, using UMAP for dimensionality reduction and highlighting 10 Pfam domains with interesting features. Pfam domains with shared sequence features are expected to be close in the dimensionally reduced space. The x and y axis correspond to the two UMAP projected dimensions. It can be seen that, for example, the related families PF07686 (blue) and PF07654 (orange) are close/ overlapped in the embeddings of CPCProt, deepGOCNN, ESM, PLUS-RNN, ProtTrans, SeqVec, TAPE and UniRep. This overlap is barely seen in the projections of BLASTp, GP, ProtVec, RBM and rawMSA.

TAPE and UniRep, these two families form a compact cluster and are very separated from the rest of the families.

The domains PF13853 (brown) and PF00001 (purple) are transmembrane, alpha helix rich molecules, sharing not only a secondary structure but also subcellular localization. Relative to these proteins enriched by alpha helices, the projections of protein sequences containing Serine protease domains PF00089 (light blue) should be located distantly since the latter domain is mainly composed of beta sheets. These two facts: PF13853 and PF00001 together or close, and at the same time well-separated from PF00089 is verified in the projections of CPCProt, deepGOCNN, ProtVec, SeqVec, TAPE and UniRep. Moreover, it is well-known that Serine protease domains are divided into two broad categories (chymotrypsin-like and subtilisin-like) which are captured by the presence of well-separated clusters of embeddings for ESM, SeqVec, UniRep and PLUS-RNN. In the other embeddings, instead, these families are widespread and mixed up with the other families.

Additionally, a large overlap between the projections corresponding to proteins containing the Homeodomain PF00046 (black) and the RNA recognition motif family PF00076 (pink) can be expected to reflect the fact that both domains are capable of binding nucleic acids. This closeness is verified for CPCProt, deepGOCNN, ESM, PLUS-RNN, ProtTrans, ProtVec, SeqVec, TAPE and UniRep. Finally, regarding separation of the embedding projections corresponding to different families, the methods that clearly separate into different clusters each domain family are ESM, PLUS-RNN, SeqVec, TAPE and UniRep.

To better evaluate, in a more quantitative way, the capacity of the different embeddings for grouping the 10 Pfam domains selected, we carried out a silhouette analysis. We measured the distances between the proteins annotated with the 10 Pfam domains using the cosine of their angular distances defined in the space of their corresponding embeddings. Figure 2 shows the silhouette profile for each method where shaded areas plot the scores obtained from protein pairs sharing the same Pfam domains. The better the clusterings the higher the silhouette scores, reaching $+1$ as maximum value. It can be seen that neither method can perfectly group all the Pfam domains in the projected space, because of the presence of negative silhouette scores in all cases. However, some methods present scores that are high, indicating that they are relatively better than others for clustering proteins according to their domains. To rank methods according to their clustering capacity, we used the silhouette coefficient of each method, which summarizes its profile for all domains and it is shown as a solid vertical line in Fig. 2. The resulting rankings indicated that the best clusterings were obtained by PLUS-RNN, ESM, UniRep and SeqVec, which is in line with the qualitative UMAP projections. To further contextualize this result, we compared the silhouette coefficients with that achieved by BLAST, indicated as a dotted vertical line. This comparison revealed that six methods were better than BLAST in this task: PLUS-RNN, ESM, UniRep, SeqVec, TAPE and ProtTrans. This result not only highlights that protein embeddings can be a powerful tool for protein domain analysis but also demonstrates the advantages of using embedding representations rather than primary protein sequences for bioinformatics tasks.

Table 2: Statistical correlation between BLASTp sequence similarities and cosine distances.

Method	**Spearman correlation**
ESM	0.66
ProtTrans	0.63
PLUS-RNN	0.57
SeqVec	0.53
TAPE	0.53
deepGOCNN	0.51
UniRep	0.49
CPCProt	0.45
ProtVec	0.42
rawMSA	0.40
GP	0.34
RBM	0.27

4.2 Protein domain prediction

Since protein domains are intimately involved in protein functionality, the correct identification of all the different domains present in a protein could give insight into its function, localization and/or biological process. Embeddings that can learn this information are highly desirable for prediction. Thus, to quantify the ability of the embedding methods to be used for functional domain prediction, that is, Pfam family prediction using the cosine similarity between vector embeddings, we set up an experiment using kNN, as indicated in Section 3.3.

The results can be seen in Figure 3 (top) where the curves for both F_1 and s^+ are displayed in the y-axis, for the corresponding k-neighbors from 1 to 10 in the x-axis. Interestingly, the predictions of all the methods share a common trend where the F_1 and recall tend to decrease and increase, respectively, as the number of k neighbors increases. This indicates that, as k increases, methods are able to predict more accurately the correct domains (higher recall) but at the cost of introducing more false positive predictions (lower F_1). It should be noticed that this trend is in accordance with the results of the previous projection analysis; both results point out that the embeddings sharing similar protein domains are found close in the vector space. Therefore, as k increases, domain inference becomes worse because of including more neighbors, which are not necessarily close in the space as indicated by the F_1. In addition, the results also show that the predictive performance varies according to the method used. The ESM method is shown to be noticeably above the rest, regardless of the number k chosen. This method achieved an average F_1 of 0.70 when considering only the first neighbor. An excellent result when compared with the second-best, SeqVec, which has an F_1 slightly below 0.60 in the same condition.

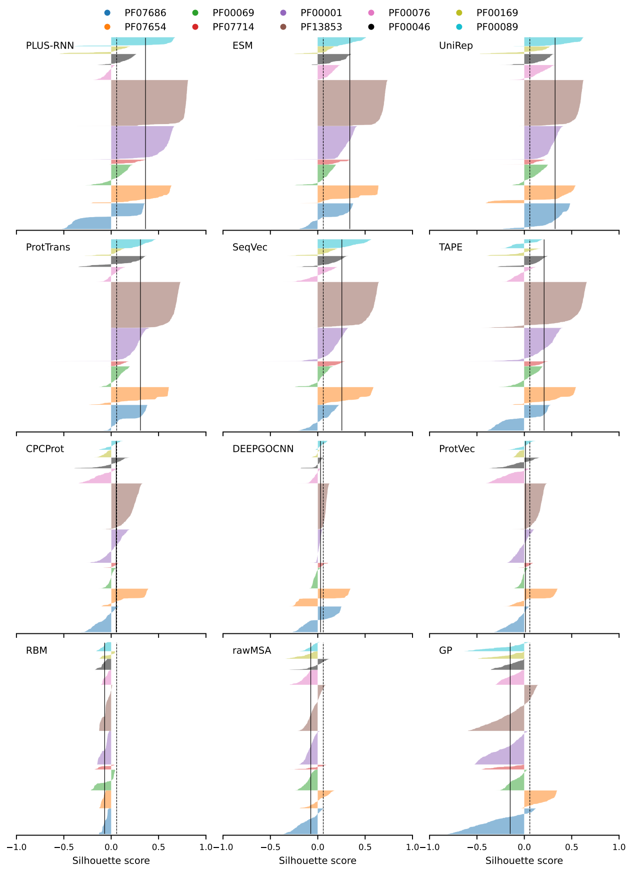

Figure 2: Silhouette profiles for clustering Pfam domains. Panels show the silhouette profiles obtained from the embedding distances of each studied method. Shaded areas plot the silhouette scores for protein pairs sharing the same (color-encoded) Pfam domain. Vertical solid lines indicate the silhouette coefficient summarizing profiles, whereas dotted lines show the coefficient of BLAST. It can be seen that in spite no method can perfectly group all the Pfam domains, some of them have high scores indicating that they are more capable of clustering proteins according to their domains.

The Pfam family domain prediction was also performed by using an MLP model, as explained in Section 3.3. These results are shown in Figure 3 (bottom), where the y-axis reports average F_1 results for the 10-fold cross-validation setup. It can be seen that ESM is the best method with $F_1 = 0.60$, clearly separated from the rest of the methods. The next best methods are SeqVec and ProtTrans with approximately $F_1 = 0.40$. Then UniRep with $F_1 = 0.30$ and PLUS-RNN around $F_1 = 0.20$. The remaining methods achieved very low performance for the Pfam family domain prediction task using an MLP model. Notably, this result also shows that the predictive performance of ESM is better than that of BLASTp-based predictions, demonstrating that embedding representations for protein sequences are able to encode protein domains faithfully.

When performing a more strict version of this experiment with similarities ¡ 60%, 40% and 20% PIDE between train and test sets, the overall behavior and differences among methods remained but the F_1 score of each method was reduced significantly as the problem became harder, which was expected, as can be seen in Supplementary Figure S2. This drop in performance mainly obeys to protein annotations that are representative in the test set but absent in the training set due to the severity of the partitioning.

In addition, we also repeated the experiment by partitioning training data by using single-linkage hierarchical clustering on the GO annotation similarity between proteins according to the Jaccard distance. That is, the partitions based on GO similarities were used to train and evaluate an MLP on the Pfam prediction task. The results showed that such data partitioning had almost no impact on predicting Pfam domains (Supplementary Figure S3), suggesting that GO annotations and Pfam domains are not strongly correlated.

These experiments of Pfam and GO terms predictions were also performed in 2 other non-model organisms: *Rattus norvegicus* and *Mycobacterium tuberculosis H37Rv*. This way the comparative study includes organisms equidistant on a scale from very-well to not-well annotated. The results (Supplementary Figure S4) show that, as expected, the performances are much lower when the species is not well-annotated, but very interestingly the same trends than for *H. sapiens* regarding best and worst groups of performing methods are observed.

4.3 Ontology-based Protein Function Prediction

To test the methods for GO terms prediction in each sub-ontology using the cosine similarity between vector embeddings, we set up an experiment using kNN and an MLP model, as explained in Section 3.3, whose results are shown in Figure 4 for each sub-ontology from top to bottom: BP, CC and MF, respectively. For each sub-ontology, column A shows the curves for F_1 on the y-axis, for the corresponding k-neighbors from $k = 1$ to $k = 10$ indicated on the x-axis; Column B shows the violin plot for the MLP predictions.

Looking at the F_1 results in BP with kNN (top, column A), it can be clearly seen that ESM is the best method in this task, with just one neighbor. The score for ESM at $k = 1$ is $F_1 = 0.35$. At increasing k, the F_1 decreases substantially, and for all methods, because there is a large amount of false positives predicted. The worst methods here are rawMSA and RBM. Analyzing s^+ in BP (Supplementary Figure S5), it can be seen that increasing the number of neighbors produces an increment in the recall value, allowing for example the recovery of more than 80% of the true GO terms on average in the case of ESM for $k = 10$. Here again, the worst methods can reach only up to a 50% recall in the best case. The corresponding clustering coefficients for the kNN (Supplementary Figure

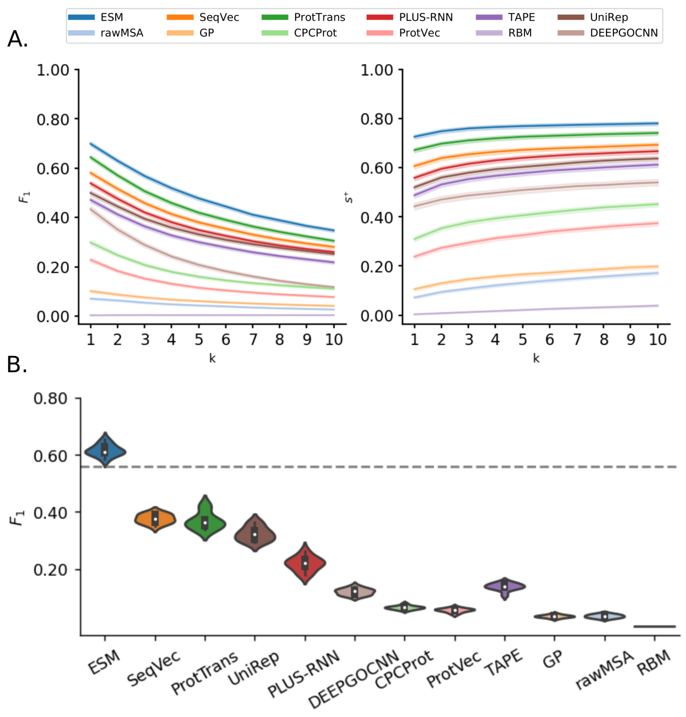

Figure 3: Pfam domain prediction from protein sequence embeddings. A) Results obtained with the kNN approach. On the y axis, the performance measurements F_1 and s^+ vary depending on the number of k neighbors selected. B) Performance of each method when used to train an MLP. The violin plots show the mean F_1 scores from the 10 fold cross-validation.

S6) show the same trend as s^+ for all methods. However, as depicted in the F_1 curve, there is a price to pay for a high recall, because together with a large number of neighbors a large number of unrelated terms are recovered, which introduce several FP predictions.

In the case of the kNN results for CC (middle, column A) and MF (bottom, column A) sub-ontologies, the same trend as with BP can be seen, although with much higher scores. In CC the maximum $F_1 = 0.55$ is reached by ESM with $k = 1$, followed by ProtTrans, SeqVec, PLUS-RNN, UniRep, deepGOCNN, CPCProt and GP, being again rawMSA and RBM the worst methods. In MF the maximum $F_1 = 0.50$ for ESM with $k = 1$, with all the other methods below it in the same decreasing order as in CC. Similarly to before, a large s^+ score can be reached (Supplementary Figure S5), for example of 0.90 in CC, at the largest k, but at the cost of a very low F_1. The same is observed for the corresponding clustering coefficients (Supplementary Figure S6).

In summary, the most accurate predictions can be seen when trying to predict the GO terms of the CC sub-ontology followed by the MF and BP

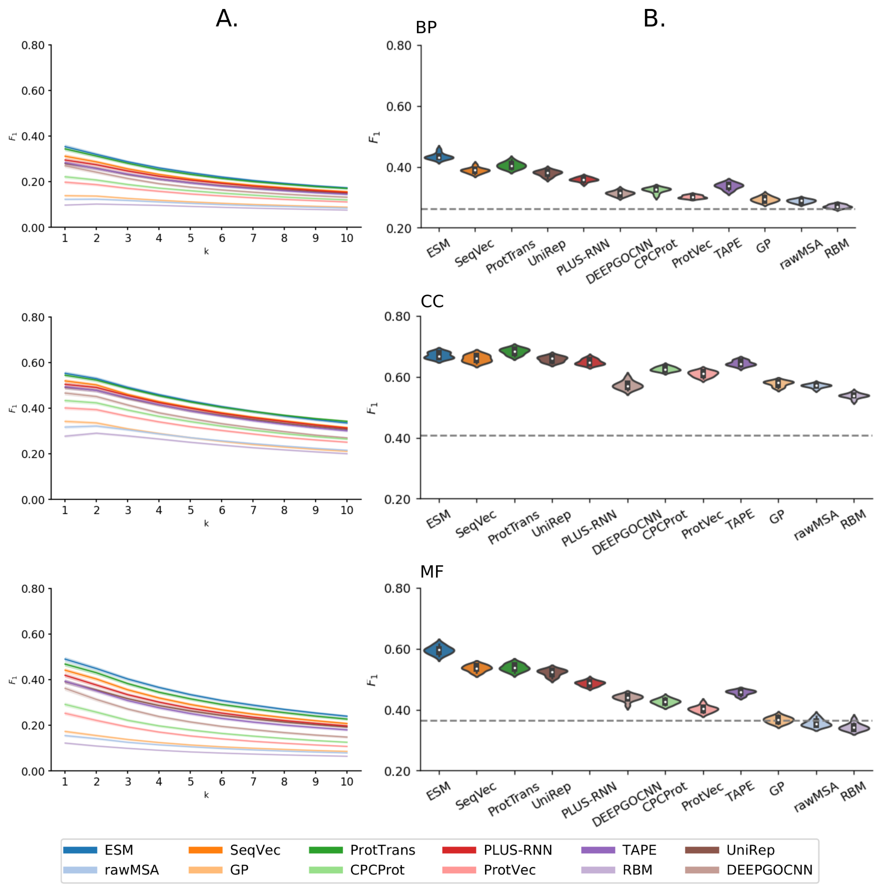

Figure 4: Ontology-based predictions. For each sub-ontology: protein sequence embeddings were used to predict GO terms observing the impact of modifying the k number of nearest neighbors with kNN (A). The violin plots (B) show the performance of each method for predicting GO terms with an MLP. BLASTp baseline in dotted line.

in that order. This difference in predicting capabilities can be seen also in the CAFA3 results across all methods and it seems to be related to how clearly motifs are encoded in protein sequences. For example, signal peptides that lead proteins to the different places within the cell (i.e. CC) are usually well conserved across kingdoms and thus are more easily detected by methods, whereas the role of a protein in a given biological process depends on many other factors, not only protein sequence information.

The same function prediction task was performed by using an MLP model, as explained in Section 3.3, whose results are shown as violin plots Figure 4 in terms of the F_1 scores obtained for each sub-ontology. In comparison to kNN, the prediction capabilities of all the methods show an increase when an MLP is used, mainly on the worst-performing methods (RBM and rawMSA). This can be explained by the fact that an MLP is not only more powerful than a simple kNN, but also because its parameters are trainable. The number of proteins varies between sub-ontologies depending on the annotation of each protein. The tendency of ESM outperforming any other method remains, even though the distance shortens regarding other methods with good performance such as ProtTrans, SeqVec or UniRep, the difference is still significant (pval¡0.001 in a binomial test without ties).

The Friedman test and the CD diagram (Supplementary Figure S7) on the BP sub-ontology results showed that F_1 value for ESM relative to all other methods is statistically significant and the best method for GO terms prediction, while ProtTrans, SeqVec, UniRep and PLUS-RNN are not statistically different amongst each other, and rawMSA and RBM are the worst here. In the case of CC sub-ontology, there are no significant differences among methods, with ProtTrans and ESM the best ones. On the MF sub-ontology, again ESM is the best and well-separated from the other methods while ProtTrans, SeqVec and UniRep are not statistically different amongst each other. In contrast to the Pfam results, almost all the methods show better performance than BLASTp-based predictions. Since protein sequence embeddings are built from large protein databases, they are better in capturing additional information beyond the primary sequence. Hence, our results indicate that, in contrast to the primary sequence information which is exclusively used for BLASTp predictions, the additional information that embeddings are able to capture shows to be a relevant factor to boost predictive performance on the GO predictive task. These experiments were also performed in 2 other non-model organisms: *Rattus norvegicus* and *Mycobacterium tuberculosis H37Rv*. The results (Supplementary Figure S8) show that the same trends in performance among methods are observed. When using the partitioning with a much harder 20% identity cutoff between training and testing sets, an expected slight drop in F_1 score can be seen across every embedder. Although, the relative performance between the methods and the group of good performing and worst performing methods remains the same (Supplementary Figure S9).

We repeated this experiment by using an alternative partitioning of the training data by using single-linkage hierarchical clustering on the Pfam annotation similarity between proteins calculated with the Jaccard distance. That is, the partitions based on Pfam similarities were used

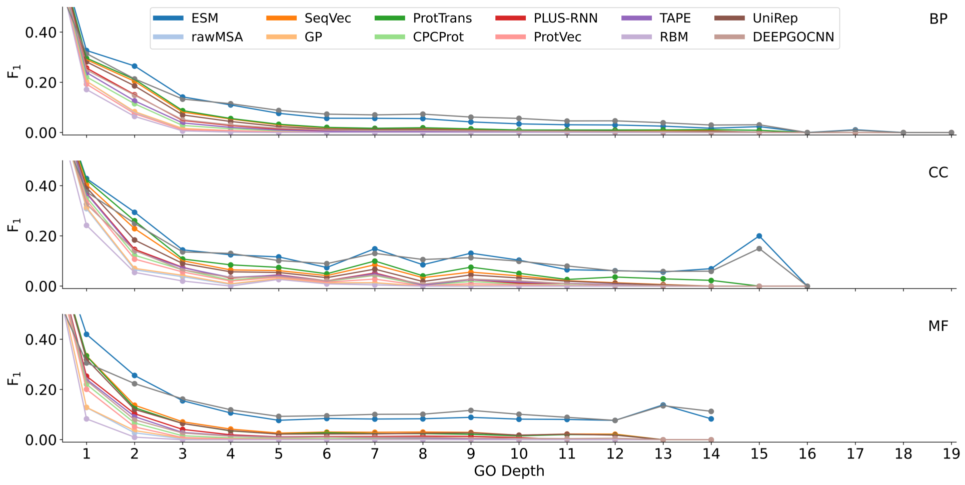

Figure 5: Ontology-based protein function prediction detail: for each sub-ontology, curves plot the predictive performance of each embedding method as a function of the term depth. The predictive performance is the F_1 score averaged over all the terms at a given depth. Baseline BLASTp predictions shown in gray.

to train and evaluate an MLP on the GO prediction task. The results (Supplementary Figure S10) showed that such data partitioning had a moderate impact on the capacity of embeddings for predicting GO terms from CC but strong for terms from the other two sub-ontologies. This points out that protein domains encode crucial information for predicting BP and MF, which is expected as they are partly governed by the specific electro-chemical properties of protein domains. In contrast, the low impact on predicting CC localizations is also expected because they are determined by transit-peptide signals which are not represented as Pfam domains.

Finally, to better and more accurately quantify the performance of the trained MLPs for predicting protein annotations as a function of the depth of the GO terms in each sub-ontology. In protein function tasks, the biological concepts represented by deeper GO terms are more specific in comparison to their ancestor terms. Thus, accurate predictions for deeper terms can be more informative for diverse bioinformatics problems, such as predicting protein-protein interactions in terms of their protein function similarities. Figure 5 plots the in-depth predictive performance of each representation method measured in terms of F_1 obtained by the same MLP as for Figure 4, which shows a flattened performance for each method. Here, the x-axis ranges the different depth values, while the y-axis represents the average F_1 score calculated over all the GO terms at a given depth. Because all the methods achieved the maximal F_1 score (1.0) at the root term of each sub-ontology (GO depth=0), these results were excluded to allow a better visualization for more informative GO terms

(GO depth¿0). By analyzing the results obtained for each sub-ontology, the BP terms were the most difficult to predict for all the methods, because they show the lowest predictive performances in comparison to the other sub-ontologies. This is in agreement with our previous results and those reported in the CAFA 3 challenge [56], suggesting that correct predictions of BP terms require not only sequence information but also additional sources of information. On the other hand, the F_1 scores of the CC and MF terms were similar to each other but higher than those of the BP terms, which is expected as CC and MF functions are usually represented as well conserved patterns in protein sequences. The predictive performances decrease as the GO depth increases on the three sub-ontologies. Although two exceptions can be seen, at depths 15 and 13-14 on CC and MF respectively, due to the fact that these cases only have 1-3 GO terms, resulting in highly biased averages. In agreement with our previous analysis, the results also show ESM as the best method for predicting protein functions, which is followed by ProtTrans, SeqVec, PLUS-RNN and UniRep. Moreover, in comparison to all the other methods, ESM yielded the best predictions, being very close to the BLASTp performance, at almost all the depth values across the three sub-ontologies. In particular, it is very interesting to see how, in this very deep analysis, ESM in the prediction of GO terms for MF shows a quite stable performance in the intermediate levels, being almost as high as in the first ones at the deepest levels. This is a remarkable result since those are the more interesting and important GO levels. The corresponding statistical test is shown in Supplementary Figure S11 where it can be seen that for BP, almost all methods are significantly different among them, with BLASTp and ESM the best ones followed by ProtTrans and SeqVec. In CC and MF, it is much more difficult to distinguish among methods.

To further analyze the performance of the embeddings from the perspective of learning, we investigated the confusion matrix derived from the GO terms predicted by each embedding and the true GO terms annotating the training proteins, which were used as classes. Analyzing the confusion matrices allowed us to see in detail how the performance was affected in small classes (i.e., low annotation frequency terms). We have calculated the confusion matrix per method for each GO term separately. Next, we computed the F_1 score from each confusion matrix and compared it against the frequency of the term according to protein annotations used as training data. The results (Supplementary Figure S12) show that, regardless of the sub-ontology, high F_1 scores are generally obtained for the large classes (highly frequent terms) while the small classes (low frequency terms) have a very variable range of F_1 scores and worst performance depending on the specific GO term. These analyses corroborated the observed tendency shown in Figure 5, where all the methods make better predictions for GO terms that are frequently found in the training set, which commonly correspond to those found close to the root of the GO.

4.4 Comparison of BLAST vs per-protein and per-residue embeddings

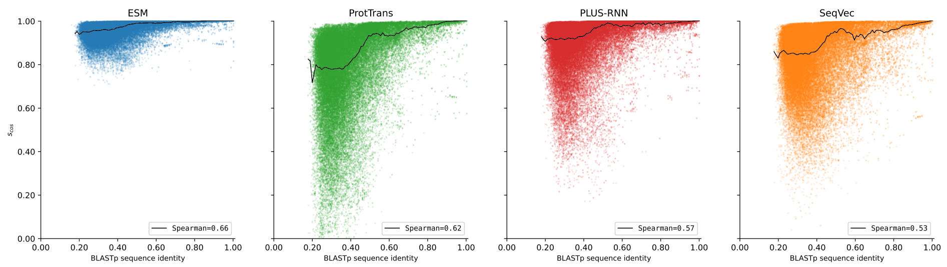

Figure 6: Sequence similarity encoded by full embeddings. Points depict protein pairs whose cosine similarities s_{cos} are calculated from per-protein embeddings. Sequence BLASTp similarities (x-axis) versus embeddings s_{cos} similarities (y-axis). Legends show Spearman correlations (the higher the better).

We investigated whether protein embeddings were able to preserve, in their own space, protein distances measured according to the sequence similarities calculated by BLASTp [2]. To this aim, we compared the sequence similarities between protein sequences obtained with BLASTp versus the cosine similarity between their corresponding per-protein embeddings, for all methods here evaluated.

The resulting comparisons for the top-4 embeddings are shown in Figure 6 (full results in Supplementary Figure S13), where point clouds depict protein pairs whose coordinates show their BLASTp (x-axis) and cosine s_{cos} (y-axis) similarities, and colors (blue for ESM, green for ProtTrans, red for PLUS-RNN and orange for SeqVec) indicate per-protein representations. The results show that for protein pairs with BLASTp scores higher than 80%, the cosine similarity of their corresponding embeddings was also high for almost all the embedding methods. This indicates that proteins sharing high sequence identity are represented by embeddings located relatively close to each other, that is, with cosine similarities close to 1. It can also be seen that the cosine s_{cos} similarity is much less variable than the sequence similarities calculated by BLASTp, leading to a poor correlation between both similarities and, thus, s_{cos} has a low capability of discriminating proteins. In fact, sequences with low sequence similarity have $s_{cos} > 0.80$ for most embedding methods. Most protein pairs obtain cosine similarities between 0.95 and 1.0 regardless of its BLASTp sequence similarity, indicating the embeddings have low discrimination power among dissimilar sequences.

To further quantify the latter correlation, we ranked embedding methods according to their capability for preserving BLASTp sequence similarities. To this aim, point clouds were summarized as curves by averaging the scores over evenly spaced intervals of BLASTp sequence similarities. Next, the Spearman correlation coefficient [33] was calculated from each resulting curve, to quantitatively measure the correlation between the BLASTp and cosine values.

The results for each method here evaluated are shown in Table 2, with correlation coefficients ordered in descending order from high to low values. The results indicate that ESM and ProtTrans obtained the largest coefficients, 0.66 and 0.63 respectively, indicating that their embeddings are the best in preserving the BLASTp sequence similarities. On the other hand, GP and RBM showed the lowest coefficients, indicating that their embeddings are not good for capturing BLASTp sequence similarities. In general, these results reveal a moderate overall correlation for the other methods between the BLASTp sequence similarities and cosine similarities. These results are consistent with previous observations for a specific embedding method [12], however, our comparisons using 12 embedding methods revealed that it is a common behavior rather than a method-specific phenomenon. This suggests that up-to-date protein vector representations are unable to fully preserve specific protein sequence information, probably because the embedding methods collapse the whole protein sequence information into a single low-dimensional vector. This is a very interesting finding since modern methods used for AFP are generally based on protein representations, and according to our experimental data, this implies that there could be difficulties in predicting protein functions governed by fine-grained sequence elements.

The results obtained above showed that the classifiers trained with protein embeddings are unable to make good predictions for the deeper GO terms, which are the most interesting ones. This suggests that protein sequence embeddings are not encoding fine-grain sequence information, which might be preventing classifiers from predicting specific protein functions. This lack of fine-grain information in embeddings can be produced because they represent proteins as a single vector usually built by reducing per-residue information.

Therefore, we evaluated the impact of using reduced (per-protein) embeddings versus per-residue representations, on preserving protein distances, measured as sequence similarities calculated by BLASTp. To this aim, the protein similarity between protein pairs was compared with the cosine similarity calculated from their corresponding per-residue embeddings. Since variable-length embeddings are obtained when using per-residue embeddings, the experiment was conducted on a subset of 1,506 protein sequences with very similar lengths, ranging from 309 to 365 residues, that were homogenized by adding gaps on their C-terminal ends. Each protein was then represented as a long vector by flattening the per-residue embeddings. As a control, the cosine similarities were also calculated for the same representations but reduced per-protein, to be able to conduct a comparative analysis between both representations.

The results of the experiment are shown in Figure 7 for the best methods ESM, ProtTrans and SeqVec, respectively (full results in Supplementary Figure S14). In the figure, each point depicts a protein pair whose coordinates show their BLASTp (x-axis) and cosine s_{cos} (y-axis) similarities, and colors (blue for ESM, green for ProtTrans and orange for SeqVec) indicate per-residue and reduced (gray) representations, respectively. This figure shows that, regardless of the method, the use of per-residue embeddings improves significantly the discrimination of protein pairs with low sequence similarity. Note that such level of discrimination

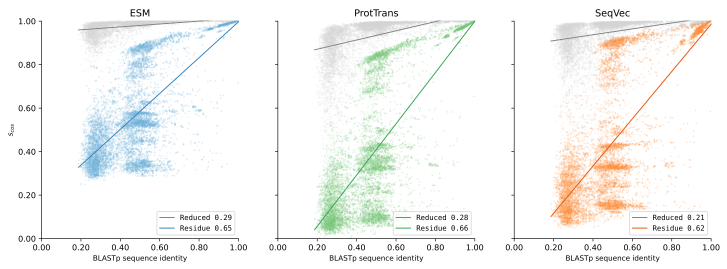

Figure 7: Sequence information encoded by per-residue embeddings. Points depict protein pairs whose cosine distances s_{cos} are calculated from reduced (gray) and per-residue (non-gray) embeddings, respectively. Legends show coefficients of determination (the higher the better) calculated when fitting a linear regression for each type of embedding. Per-residue embeddings correlate more with sequence similarity than per-sequence embeddings.

is not achieved when using the reduced embeddings, where all protein pairs obtain cosine similarities between 0.95 and 1.0 (the gray points at the top of the figure) regardless of its BLASTp sequence similarity. To quantify the observed tendencies, a linear regression was performed for each type of embedding. Regressions revealed that cosine and BLASTp similarities are more strongly and positively correlated when per-residue embeddings are used. The contribution of using per-residue information is also observed on their coefficients of determination, calculated from regressions, which are higher than those obtained by their reduced counterparts: for ESM 0.65 when using a per-residue embedding versus 0.29 when using the reduced version; for ProtTrans 0.66 when using a per-residue embedding versus 0.28 when using the reduced version; for SeqVec 0.62 versus 0.21. Note, however, that although the per-residue embeddings are better than the per-protein embeddings for protein representations, the former are considerably larger, occupying a substantial amount of computer memory. Nevertheless, the size of the per-residue representations can be reduced by using lower floating point precision. We experimentally evaluated this by halving the float-precision format of the per-residue embeddings and measured again their level of correlation with the pairwise BLASTp similarities (Supplementary Figure S15). The results suggest that no significant loss of correlation was observed, indicating that lowering floating point precision is a viable solution for reducing embedding size.

5 Discussion

Nowadays, in spite of the recent breakthrough of alpha-fold's structure-based prediction of protein annotation, sequence-based prediction remains the most widely-used approach. This is especially true for novel species, since their protein sequences are the most common available information. As a consequence, many learned protein sequence representation methods, named embeddings, have appeared in the last 6 years for tackling the problem of automated protein function prediction given the ever-growing availability of protein sequences.

Most of the representation methods reviewed here have approached the problem with supervised machine learning, many of them through CNNs and recurrent networks of varying architectures trained on UniProt and Pfam data. The few proposals based on unsupervised models are all based on Transformers, the newest and highly performant deep learning architecture, which is showing breakthrough results in a large number of practical problems.

The different comparisons among embedding methods included in this study have shown that ESM, ProtTrans and SeqVec were the best methods for most bioinformatics tasks evaluated here: protein similarity in the embedding space, inference of protein domains, prediction of GO terms and BLAST similarity correlation. In particular, for GO terms predictions at different depth levels of the structural hierarchy of the GO, it was remarkable to see how ESM achieved a high performance at almost all levels. This is relevant because accurate predictions for deep GO terms are of high interest for different fields in biology, since they represent very specific protein functions.

It is worth mentioning the difference in results with a recent review on protein embeddings [**?**], which used a different benchmark of human proteins from UniProtKB/Swiss-Prot annotated with UniProt-GOA of date 2019 for testing, while we used a larger set of proteins from 3 species of the standard CAFA3 challenge. Additionally, while the mentioned work used SVM we used kNN and MLP for the predictions. Finally, the set of methods evaluated was different.

Regarding the per-protein versus per-residue preferable representation of the embedding methods, the experiments revealed that the use of per-residue embeddings leads to embedding similarities that more faithfully reflect sequence similarities. Since a positive correlation was observed for the per-residue embeddings of diverse methods, the benefits of using per-residue information seems to be a general rather than a method-specific effect. To the best of our knowledge, per-residue information is not yet directly exploited by existing predictive methods. Hence, it can be a key factor to be used by future methods to further boost predictive power. However, per-residue embeddings present some disadvantages, such as having many vectors as the number of residues composing a protein, increasing the size of the embedding significantly and requiring more computational resources to process it. Besides considerably increasing the representation size, this also leads to variable-length representations for proteins, which can introduce new processing challenges in downstream tasks.

Nevertheless, these challenges posed from using per-residue embeddings can be partially faced by using models capable of processing variable-length data, such as recurrent neural networks, in order to reduce the per-residue information into fixed-length representations per protein in a more data-driven way. It was also found that per-residue embedding size can be effectively reduced by halving the point floating precision used for computationally representing them because this has no impact on the final accuracy. Altogether, these analyses allow us to state that the use of per-residue information is the next big challenge in protein representation learning and where future efforts should be directed in this area, in order to be able to capture finer-grained information encoded throughout the full length of the protein sequences.

6 Conclusions

In this study, we provided a comprehensive review and experimental comparison of 12 protein sequence representation learning methods that have appeared in the last 6 years. We have compared them on several and very concrete bioinformatics tasks: (i) ability of protein embeddings to preserve protein similarities according to BLAST sequence similarities; (ii) ability of protein embeddings to encode similar molecular features in a common region in the multidimensional embedding space; (iii) ability of the embedding methods to be used for functional domain prediction; (iv) testing of the protein representations for automatic function prediction, that is, GO terms prediction in each sub-ontology using the similarity between vector embeddings.

We also compared the reduced per-protein versus the per-residue representation available for some embedding methods, analyzing the advantages and disadvantages of each representation approach. We hope the results and the discussions of this study can help the community to select the most adequate machine learning-based representation technique for protein data representation, according to the bioinformatics task at hand.

Key points

- We provide a comprehensive review and experimental comparison of 12 protein representation learning methods from the last 6 years.
- Each method has been explained briefly and compared experimentally on the same data set of protein sequences.
- The protein embedding have been compared in 3 very concrete bioinformatics tasks: (i) determining protein sequence similarity in the embedding space; (ii) inferring protein domains; and (iii) predicting ontology-based protein functions.
- We hope this study can help the community to better understand these newly provided protein representations, in order to select the most adequate according to the bioinformatics task at hand.

Funding

This work was supported by Agencia Nacional de Promocion Cientifica y Tecnologica (ANPCyT) [PICT 2018 3384] and UNL [CAI+D 2020 115].

Conflict of Interest: none to be declared.

Biographies

E. Fenoy is a bioinformatician with experience in immunoinformatics and proteomics. He currently holds a postdoctoral position at the Research Institute for Signals, Systems and Computational Intelligence, sinc(i), Santa Fe, Argentina.

A. Edera is a postdoctoral fellow in the Bioinformatics lab at sinc(i) Institute, Santa Fe, Argentina. His research includes computational and statistical aspects of biological problems with special focus on plants.

G. Stegmayer is Professor in the Department of Informatics at Universidad Nacional del Litoral (UNL), and Independent Researcher at the sinc(i) Institute, National Scientific and Technical Research Council (CONICET), Argentina. She is the Leader of the Bioinformatics lab at sinc(i). Her current research interest involves machine learning, data mining and pattern recognition in bioinformatics.

sinc(i) - Research Institute for Signals, Systems and Computational Intelligence. Research at sinc(i) aims to develop new algorithms for machine learning, data mining, signal processing and complex systems, providing innovative technologies for advancing healthcare, bioinformatics and human-computer interfaces. The sinc(i) was created and is supported by the two major institutions of highest education and research in Argentina: the National University of Litoral (UNL) and the National Scientific and Technical Research Council (CONICET).

References

[1] Alley, E. C., Khimulya, G., Biswas, S., AlQuraishi, M., and Church, G. M. (2019). Unified rational protein engineering with sequence-based deep representation learning. *Nature Methods*, **16**(12), 1315–1322.

[2] Altschul, S. F., Madden, T. L., Schäffer, A. A., Zhang, J., Zhang, Z., Miller, W., and Lipman, D. J. (1997). Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. *Nucleic Acids Research*, **25**(17), 3389–3402.

[3] Asgari, E. and Mofrad, M. R. K. (2015). Continuous distributed representation of biological sequences for deep proteomics and genomics. *PLOS ONE*, **10**(11), 1–15.

[4] Becht, E., McInnes, L., Healy, J., Dutertre, C.-A., Kwok, I. W., Ng, L. G., Ginhoux, F., and Newell, E. W. (2019). Dimensionality reduction for visualizing single-cell data using umap. *Nature biotechnology*, **37**(1), 38–44.

[5] Bepler, T. and Berger, B. (2018). Learning protein sequence embeddings using information from structure. In *International Conference on Learning Representations*.

[6] Bishop, C. (2014). Bishop-pattern recognition and machine learning-springer 2006. *Antimicrob. Agents Chemother*, pages 03728–14.

[7] Bishop, C. M. (2007). *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer, 1 edition.

[8] Consortium, T. G. O. (2018). The Gene Ontology Resource: 20 years and still GOing strong. *Nucleic Acids Research*, **47**(D1), D330–D338.

[9] Consortium, U. (2019). Uniprot: a worldwide hub of protein knowledge. *Nucleic acids research*, **47**(D1), D506–D515.

[10] Cook, G. (2000). Immunobiology: The immune system in health and disease (4th edn) by ca janeway, p. travers, m. walport and jd capra. *Immunology today*, **21**(4), 201.

[11] Dalkiran, A., Rifaioglu, A. S., Martin, M. J., Cetin-Atalay, R., Atalay, V., and Dogan, T. (2018). Ecpred: a tool for the prediction of the enzymatic functions of protein sequences based on the ec nomenclature. *BMC bioinformatics*, **19**(1), 334–334.

[12] Dallago, C., Schütze, K., Heinzinger, M., Olenyi, T., Littmann, M., Lu, A. X., Yang, K. K., Min, S., Yoon, S., Morton, J. T., and Rost, B. (2021). Learned embeddings from deep learning to visualize and predict protein sets. *Current Protocols*, **1**(5), e113.

[13] Demšar, J. (2006). Statistical comparisons of classifiers over multiple data sets. *J. Mach. Learn. Res.*, **7**, 1–30.

[14] Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

[15] ElGebali, S., Mistry, J., Bateman, A., Eddy, S. R., Luciani, A., Potter, S. C., Qureshi, M., Richardson, L. J., Salazar, G. A., Smart, A., Sonnhammer, E. L., Hirsh, L., Paladin, L., Piovesan, D., Tosatto, S. C., and Finn, R. D. (2018). The pfam protein families database in 2019. *Nucleic Acids Research*, **47**(D1), D427–D432.

[16] Elnaggar, A., Heinzinger, M., Dallago, C., Rehawi, G., Yu, W., Jones, L., Gibbs, T., Feher, T., Angerer, C., Steinegger, M., Bhowmik, D., and Rost, B. (2021). Prottrans: Towards cracking the language of lifes code through self-supervised deep learning and high performance computing. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **1**(1), 1–10.

[17] Graves, A., Fernández, S., and Schmidhuber, J. (2005). Bidirectional lstm networks for improved phoneme classification and recognition. In *International conference on artificial neural networks*, pages 799–804. Springer.

[18] Hastie, T., Tibshirani, R., and Friedman, J. (2009). *The elements of statistical learning: data mining, inference and prediction*. Springer, 2 edition.

[19] Heinzinger, M., Elnaggar, A., Wang, Y., Dallago, C., Nechaev, D., Matthes, F., and Rost, B. (2019). Modeling aspects of the language of life through transfer-learning protein sequences. *BMC Bioinformatics*, **20**(1).

[20] Hinton, G. E. (2012). A practical guide to training restricted boltzmann machines. In *Lecture Notes in Computer Science*, pages 599–619. Springer Berlin Heidelberg.

[21] Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, **9**(8), 1735–1780.

[22] Jain, A. and Kihara, D. (2018). Phylo-PFP: improved automated protein function prediction using phylogenetic distance of distantly related sequences. *Bioinformatics*, **35**(5), 753–759.

[23] Keskin, O., Tuncbag, N., and Gursoy, A. (2016). Predicting protein–protein interactions from the molecular to the proteome level. *Chemical Reviews*, **116**(8), 4884–4909.

[24] Kingma, D. P. and Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

[25] Klopfenstein, D., Zhang, L., Pedersen, B. S., Ramírez, F., Vesztrocy, A. W., Naldi, A., Mungall, C. J., Yunes, J. M., Botvinnik, O., Weigel, M., *et al.* (2018). Goatools: A python library for gene ontology analyses. *Scientific reports*, **8**(1), 1–17.

[26] Kulmanov, M. and Hoehndorf, R. (2019). DeepGOPlus: improved protein function prediction from sequence. *Bioinformatics*, **36**(2), 422–429.

[27] Kulmanov, M. and Hoehndorf, R. (2020). Deepgoplus: improved protein function prediction from sequence. *Bioinformatics*, **36**(2), 422–429.

[28] Le, Q. and Mikolov, T. (2014). Distributed representations of sentences and documents. In E. P. Xing and T. Jebara, editors, *Proceedings of the 31st International Conference on Machine Learning*, volume 32 of *Proceedings of Machine Learning Research*, pages 1188–1196.

[29] LeCun, Y., Bengio, Y., and Hinton, G. (2015). Deep learning. *Nature*, **521**(7553), 436–444.

[30] Littmann, M., Heinzinger, M., Dallago, C., Olenyi, T., and Rost, B. (2021). Embeddings from deep learning transfer go annotations beyond homology. *Sci Rep*, **11**(1160).

[31] Lu, A. X., Zhang, H., Ghassemi, M., and Moses, A. (2020). Self-supervised contrastive learning of protein representations by mutual information maximization.

[32] Makrodimitris, S., van Ham, R. C. H. J., and Reinders, M. J. T. (2020). Automatic gene function prediction in the 2020's. *Genes*, **11**(11).

[33] McDonald, J. H. (2009). *Handbook of biological statistics*, volume 2. sparky house publishing Baltimore, MD.

[34] McInnes, L., Healy, J., and Melville, J. (2018). Umap: Uniform manifold approximation and projection for dimension reduction.

[35] Mikolov, T., Sutskever, I., Chen, K., Corrado, G., and Dean, J. (2013a). Distributed representations of words and phrases and their compositionality. In *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'13, page 3111–3119, Red Hook, NY, USA. Curran Associates Inc.

[36] Mikolov, T., Chen, K., Corrado, G., and Dean, J. (2013b). Efficient estimation of word representations in vector space. In Y. Bengio and Y. LeCun, editors, *1st International Conference on Learning Representations, ICLR 2013, Scottsdale, Arizona, USA, May 2-4, 2013, Workshop Track Proceedings*.

[37] Min, S., Park, S., Kim, S., Choi, H.-S., Lee, B., and Yoon, S. (2021). Pre-training of deep bidirectional protein sequence representations with structural information. *IEEE Access*, **9**, 123912–123926.

[38] Mirabello, C. and Wallner, B. (2019). rawMSA: End-to-end deep learning using raw multiple sequence alignments. *PLOS ONE*, **14**(8), e0220182.

[39] Moult, J., Fidelis, K., Kryshtafovych, A., Schwede, T., and Tramontano, A. (2018). Critical assessment of methods of protein structure prediction (casp)—round xii. *Proteins, structure, function, and bioinformatics*, **86**(S1), 7–15.

[40] Nair, V. and Hinton, G. E. (2010). Rectified linear units improve restricted boltzmann machines. In *Icml*.

[41] Nguyen, S., Li, Z., and Shang, Y. (2017). Deep networks and continuous distributed representation of protein sequences for protein quality assessment. In *2017 IEEE 29th International Conference on Tools with Artificial Intelligence (ICTAI)*, pages 527–534.

[42] Peters, M. E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., and Zettlemoyer, L. (2018). Deep contextualized word representations.

[43] Rao, R., Bhattacharya, N., Thomas, N., Duan, Y., Chen, X., Canny, J. F., Abbeel, P., and Song, Y. S. (2019). Evaluating protein transfer learning with TAPE. *CoRR*, **abs/1906.08230**.

[44] Rasmussen, C. and Williams, C. (2006). *Gaussian Processes for Machine Learning*. Adaptive Computation and Machine Learning. MIT Press, Cambridge, MA, USA.

[45] Rifaioglu, A., Dogan, T., Martin, M., Cetin-Atalay, R., and Atalay, M. V. (2019). Deepred: Automated protein function prediction with multi-task feed-forward deep neural networks. *Scientific Reports*, **9**, 1–10.

[46] Rives, A., Meier, J., Sercu, T., Goyal, S., Lin, Z., Guo, D., Ott, M., Zitnick, C. L., Ma, J., and Fergus, R. (2020). Biological structure and function emerge from scaling unsupervised learning to 250 million protein sequences.

[47] Steinegger, M., Mirdita, M., and Soding, J. (2019). Protein-level assembly increases protein sequence recovery from metagenomic samples manyfold. *Nature Methods*, **16**(7), 603–606.

[48] Tschannen, M., Djolonga, J., Rubenstein, P. K., Gelly, S., and Lucic, M. (2020). On mutual information maximization for representation learning. In *8th International Conference on Learning Representations (ICLR)*.

[49] Tubiana, J., Cocco, S., and Monasson, R. (2019). Learning protein constitutive motifs from sequence data. *eLife*, **8**.

[50] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. (2017). Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

[51] Wang, G. and Dunbrack, R. L. (2003). PISCES: a protein sequence culling server. *Bioinformatics*, **19**(12), 1589–1591.

[52] Weiss, K., Khoshgoftaar, T. M., and Wang, D. (2016). A survey of transfer learning. *Journal of Big data*, **3**(1), 9.

[53] Yang, K. K., Wu, Z., Bedbrook, C. N., and Arnold, F. H. (2018). Learned protein embeddings for machine learning. *Bioinformatics*, **34**(15), 2642–2648.

[54] You, R., Zhang, Z., Xiong, Y., Sun, F., Mamitsuka, H., and Zhu, S. (2018). GOLabeler: improving sequence-based large-scale protein function prediction by learning to rank. *Bioinformatics*, **34**(14), 2465–2473.

[55] Zhang, S., Krieger, J. M., Zhang, Y., Kaya, C., Kaynak, B., Mikulska-Ruminska, K., Doruker, P., Li, H., and Bahar, I. (2021). Prody 2.0: increased scale and scope after 10 years of protein dynamics modelling with python. *Bioinformatics*, **37**(20), 3657–3659.

[56] Zhou, N., Jiang, Y., Bergquist, T. R., Lee, A. J., Kacsoh, B. Z., Crocker, A. W., Lewis, K. A., Georghiou, G., Nguyen, H. N., Hamid, M. N., Davis, L., Dogan, T., Atalay, V., Rifaioglu, A. S., Dalkıran, A., Atalay, R. C., Zhang, C., Hurto, R. L., Freddolino, P. L., Zhang, Y., Bhat, P., Supek, F., Fernández, J. M., and Gemovic, B. (2019). The CAFA challenge reports improved protein function prediction and new functional annotations for hundreds of genes through experimental screens. *Genome Biology*, **20**(1), 244–250.