Employing Machine Learning Techniques to Detect Protein Function: A Survey, Experimental, and Empirical Evaluations

Kamal Taha, *Senior Member, IEEE*
Department of Electrical Engineering and Computer Science, Khalifa University, Abu Dhabi, UAE. E-mail: kamal.taha@ku.ac.ae

***Abstract*—This review article delves deeply into the various machine learning (ML) methods and algorithms employed in discerning protein functions. Each method discussed is assessed for its efficacy, limitations, potential improvements, and future prospects. We present an innovative hierarchical classification system that arranges algorithms into intricate categories and unique techniques. This taxonomy is based on a tri-level hierarchy, starting with the methodology category and narrowing down to specific techniques. Such a framework allows for a structured and comprehensive classification of algorithms, assisting researchers in understanding the interrelationships among diverse algorithms and techniques. The study incorporates both empirical and experimental evaluations to differentiate between the techniques. The empirical evaluation ranks the techniques based on four criteria. The experimental assessments rank: (1) individual techniques under the same methodology sub-category, (2) different sub-categories within a same category, and (3) the broad categories themselves. Integrating the innovative methodological classification, empirical findings, and experimental assessments, the article offers a well-rounded understanding of ML strategies in protein function identification. The paper also explores techniques for multi-task and multi-label detection of protein functions, in addition to focusing on single-task methods. Moreover, the paper sheds light on the future avenues of ML in protein function determination.**

***Index Terms* - Machine Learning, Protein Function Predication, Protein Function prediction using ML, Neural Networks.**

I. INTRODUCTION

Proteins, as organic macromolecules, shape the structure and function of all living organisms, playing roles in biochemical reactions, signal transmissions, nutrient transport, and immune responses. Their study is essential for both biological insights and advancements in biomedical, pharmaceutical, and other fields. Recognizing a protein's function is vital in proteomic research. Functional characterization of proteins aids in understanding cellular processes, identifying genetic or protein changes linked to diseases, and innovating diagnostic and therapeutic tools. Despite the advancements in protein identification due to next-generation sequencing, there is still a notable disconnect between the sequencing of proteins and the understanding of their specific functions. This challenge in predicting protein functions (PFP) has grown due to the disparity and the limitations of traditional lab processes, leading to a rise in computational methods.

Protein function is defined through frameworks like Gene Ontology (GO), which classifies functions into molecular functions, cellular components, and biological processes, and Enzyme Commission (EC) numbers that categorize enzymes by their catalytic actions. Proteins are also grouped into functional families, such as serine proteases, based on evolutionary and structural similarities. A protein's function is intrinsically linked to its structure, where its three-dimensional shape and specific features like active sites determine interactions with other molecules. The amino acid sequence influences the protein's folding and function, with certain motifs correlating with specific activities. Protein function often depends on interactions with other proteins, DNA, RNA, or small molecules, affecting their activity, stability, and localization, exemplified by receptor proteins whose functions are contingent on ligand interactions.

Historically, protein functions were identified through lab experiments and then documented in biological databases [1, 2]. But these traditional methods, often resource-intensive and sometimes inaccurate [1, 3], haven't kept pace with the rapid growth in biological sequence data. As a result, computational annotation for proteins has become essential. Recent methodologies, using machine learning, aim to automate protein annotation [4-10]. The shift towards computational prediction facilitates quicker protein annotation [11, 12].

Machine learning for predicting protein function has evolved with techniques like Convolutional Neural Networks (CNNs) [13-16], Recurrent Neural Networks (RNNs) [17, 18-20], K-Nearest Neighbors (KNN) [21, 22, 23-25], and hybrid models like CNN-RNN [26] being developed. These methods leverage varied data sources, such as sequence-based and structure-based. With advancements in deep learning like CNN [27-30] and RNN, evaluating their potential in protein function prediction becomes vital. Recent research has proposed an innovative semi-supervised learning approach based on a Gaussian random field model [31].

The Critical Assessment of Protein Function Annotation (CAFA) underscores the role of machine learning, particularly DNNs [32]. DNNs, starting with basic input features, build complex layers of information. Notably, DNNs have been benchmarks in areas like computer vision and language processing [33-37]. Their potential is evident in bioinformatics and cheminformatics due to accessible computational resources. DNNs can be divided based on their modeling approach and structure. Recognized DNN architectures include feed-forward DNN, RNN, CNN, and more [38-40]. This paints a picture of the current state in protein function prediction.

A. Motivation and Key Contributions

Survey articles concentrating on machine learning (ML) approaches for predicting protein function are currently faced with the challenge of accurately classifying these approaches. Present classifications are overly general, missing specificity, and failing to distinguish between individual techniques. Such ambiguity can result in the mislabeling of unrelated methods and inconsistent evaluations. This paper presents an innovative taxonomy that breaks down algorithms into detailed categories and unique techniques. Featuring a three-tier structure, from the general methodological category to individual techniques, this classification system ensures a more systematic and comprehensive grouping of algorithms, aiding researchers in identifying links between various techniques.

The main objective of this article is to conduct an extensive review of ML algorithms applied in protein function prediction,

using a consistent technique, methodology sub-category, and broader methodology category. By utilizing this fresh taxonomy, scientists can more accurately compare and critique algorithms, enhancing their grasp of the strengths and weaknesses of each method. Furthermore, this categorization framework guides ongoing research, influencing the development and evaluation of novel algorithms. This contribution greatly advances the field of ML-powered protein function detection by presenting a more systematic and encompassing method of categorizing algorithms. It is hoped that the academic community will adopt this taxonomy, propelling the development of accurate algorithms. This study not only provides a comprehensive classification system for ML-based protein function prediction algorithms but also integrates **empirical and experimental evaluations** to measure the effectiveness of diverse methods.

Our empirical analysis assesses ML-driven protein function prediction techniques based on the following four distinct criteria. (1) *Foundational Principle:* Essential for distinguishing how ML methods conceptualize and address protein function detection, clarifying the suitability of different approaches like supervised versus unsupervised learning, (2) *Rationale Behind Application:* Focuses on why specific methods are selected for certain protein function detection types, highlighting their relevance and effectiveness in meeting unique challenges, (3) *Conditions for Optimal Performance:* Identifies the environmental and algorithmic conditions impacting a method's performance, guiding optimization for the best results and understanding failure scenarios, and (4) *Limitations:* Recognizes each method's limitations, aiding in informed method selection and pinpointing areas for future research to address these shortcomings.

Our experimental analysis ranks: (1) algorithms using the same specific technique, (2) different methodology sub-categories using the same overarching methodology category, and (3) distinct methodology categories. The holistic evaluation approach equips scientists with the means to distinguish subtle differences between related techniques. This facilitates the selection of the most appropriate ML method for specific protein function prediction tasks.

The paper also explores techniques for multi-task and multi-label detection of protein functions, in addition to focusing on single-task methods. Integrating a methodological taxonomy with empirical and experimental evaluations offers a deeper, more intricate comprehension of the existing ML algorithms for protein function detection. Consequently, researchers can make informed decisions regarding technique choice. This strategy represents a significant progression in protein function detection research that simplify the process of algorithm selection.

B. Our Proposed Methodology-Based Taxonomy

We categorize algorithms that use ML into five main groups: Neural Networks, Traditional Machine Learning, Ensemble Learning, Unsupervised Learning, and Attention Mechanism. Each of these groups is further classified into two levels, with each level becoming more detailed. Our methodology classification is organized as: category → sub-category → techniques. This tiered framework enables us to identify specific techniques in the final step. Fig. 1 depicts this method-based classification approach. There are multiple advantages to our classification system, including the following:

1. Structure: It offers a systematic structure that showcases survey results and helps readers follow the paper's flow.
2. Comprehensive Review: Ensuring complete coverage, our taxonomy includes all relevant methods, highlighting areas needing more exploration.
3. Technique Comparison: The categorization aids in contrasting research techniques, their pros and cons.
4. Reproducibility: Our taxonomy promotes easier replication by detailing clear technique outlines.

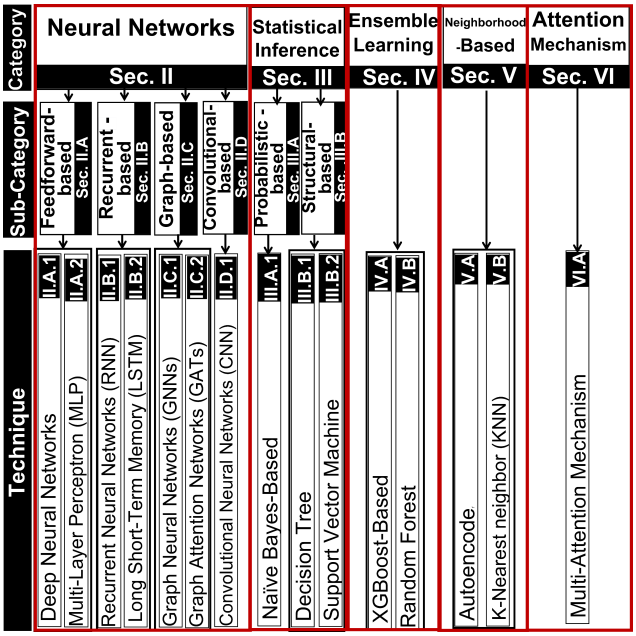


Fig. 1: Our methodology-based taxonomy that classifies the machine learning algorithms for protein function detection into the following fine-grained classes in a hierarchical manner: methodology category → methodology sub-category → methodology techniques. For each category, sub-category, technique, the figure shows the section number in the manuscript that discusses it.

II. Neural Networks Category

By learning patterns from data, neural networks can make accurate predictions regarding the function of a protein based solely on its sequence or structure.

A. Feedforward-Based Sub-Category

Feedforward neural networks (FNNs) represent one of the simpler yet effective machine learning architectures used in the prediction of protein function. The approach is relatively straightforward compared to more complex models, making it easier to implement and understand. FNNs usually take a fixed-size vector as input. This could be derived from protein sequences, such as one-hot encoded amino acids, k-mer frequencies, or feature vectors that encapsulate evolutionary or physicochemical properties. A typical FNN consists of an input layer, one or more hidden layers, and an output layer. The hidden layers use activation functions like ReLU or Sigmoid to introduce non-linearity into the model.

1) Deep Neural Networks (DNN) Technique

Rifaioglu et al. [41] introduced DEEPred, a hierarchical structure of multi-task feed-forward deep neural networks tailored for protein function prediction based on Gene Ontology (GO). DEEPred underwent thorough hyper-parameter optimizations and incorporated electronically generated GO annotations into training to assess the impact of extensive but possibly inconsistent data. Kralj et al. [42] introduced a time-sensitive multiscale biological data-based method for efficient system-wide prediction of protein functions. They experimented with various learning algorithms and found that DNN outperformed simpler models.

Yuan et al. [43] presented a DNN structure with multiple heads and ends for comprehensive protein function classification. This DNN consists of three components: the body, the multi-end, and the multi-head. The body is a deep multilayer perceptron (MLP) used for feature transformation, while the multi-end and multi-head segments handle feature integration and linear multi-label classification, respectively.

Fa et al. [44] developed a MTDNN structure to address the multi-label challenge in protein function prediction. MTDNN distinguishes between common feature representations across all GO terms and distinctive feature traits for individual terms. It utilizes two consecutive multi-layer architectures, one shared across all tasks and another dedicated to each specific task, built atop the shared layer. Tavanaei et al. [45] introduced a deep convolutional neural network (DCNN) method for identifying tertiary protein functions. Their DCNN processes feature maps of three projected images separately and classifies them using a fully connected neural network.

2) Multi-Layer Perception (MLP) Technique

Cerri et al. [46] presented the HMC-LMLP, a distinct local technique for protein function prediction using an MLP for each hierarchical tier, where results from one layer feed the next. This approach combines Back-propagation and Resilient Back-propagation MLP algorithms and uses a specialized error metric for multi-label tasks. Ashtawy et al. [47] proposed an ensemble method based on the ANN model, inspired by the Random Forests approach, but employing MLP ANNs instead of decision trees. Yavuz et al. [48] developed a method for protein secondary structure and function prediction, training data with the clonal selection algorithm (CSA) followed by classification using an MLP, showing improved results.

B. Recurrent-Based Sub-Category

Protein sequences are numerically encoded and fed into these networks to capture long-range interactions between amino acids. In the case of 3D structures, features like distance matrices are transformed into sequences for analysis. Recurrent layers may be coupled with dense classification layers, convolutional layers, or attention mechanisms to improve performance. These methods excel at capturing sequential dependencies but can be computationally intensive.

1) Recurrent Neural Networks (RNN) Technique

Cao et al. [49] redefined protein function prediction as a language translation task, introducing a protein sequence language "ProLan" and translating it into a protein function language "GOLan" using an RNN-based neural machine translation model. Liu [18] used recurrent neural networks for protein function prediction, training on amino-acid sequences from the UniProt database without additional feature extraction. Positive classes were based on matching UniProt function keywords, while negative ones used non-matching sequences.

Li et al. [19] developed GONET, a deep model combining recurrent convolutional neural networks that integrates protein sequences and PPI data. Through representation learning, GONET addresses sparsity and semantic independence, with its RNN-Attention mechanism adeptly extracting complex protein sequence features. Xia et al. [20] presented PFmulDL, combining RNNs and CNNs for protein function annotation and incorporating transfer learning to enhance accuracy. Noviello et al. [50] employed an RNN with alternating bidirectional LSTM layers for short ncRNA classification, reducing computational costs and improving resilience to sequence boundary disturbances.

2) Long Short-Term Memory (LSTM) Technique

Ranjan et al. [51] developed ProtVecGen using a unique segmentation technique and bi-directional LSTM to produce fixed-length protein vectors, enhancing function prediction. Enhanced features from varying segment sizes led to the advanced ProtVecGen-Plus. Zhang et al. [52] presented DeepGOA, a framework that predicts protein functions using protein sequences and PPI networks. This employs word2vec for sequence representation, Bi-LSTM for semantics, and a multi-scale CNN. Wekesa et al. [53] introduced LPI-DL for predicting interactions between plant lncRNA and proteins, utilizing sequence attributes and a compact LSTM. Feature selection is achieved through RFE-SVM, and sparsity via connection pruning. Shen et al. [54] used a hierarchical bidirectional LSTM with attention for predicting RNA-protein binding, optimizing various k-mer hyperparameters.

C. Graph-Based Sub-Category

Graph-based machine learning models have gained increasing attention for predicting protein function due to their ability to naturally represent biological structures and interactions. They capture the complex relationships between amino acids or residues in proteins, making them particularly effective for this task. These models are adept at learning from the intricate patterns in the 3D structures of proteins, where each node in the graph can represent an amino acid and the edges depict the interactions between them. This approach allows for a more nuanced understanding of the protein's function, leveraging the spatial and contextual information inherent in its structure.

Proteins are represented as graphs, where a node can represent an element such as amino acids, and an edge can represent spatial or sequential relationships between them. A node can also represent biological elements such as protein domains, interactions, functional annotations, sequence motifs, structural features, and post-translational modifications. This representation enables the integration of diverse biological data, facilitating a comprehensive analysis of protein functions and interactions. Edges in these graphs not only signify physical connections but can also encode functional relationships or evolutionary conserved patterns, providing a rich context for understanding the protein's role in biological processes.

These graphs may also include additional features like physicochemical properties or evolutionary information. Graph-based ML models, in addition to modeling protein relationships, often use valuable Gene Ontology (GO) data. GO's structured vocabulary defines biological functions, processes, and components, boosting protein function prediction by utilizing GO's hierarchical and semantic relationships. Incorporating GO data enables these models to align protein structures with specific biological roles, offering a more detailed and accurate function prediction. The hierarchical nature of GO allows for the prediction of protein functions at different levels of specificity, from broad biological processes to precise molecular activities, enhancing the model's ability to capture the full spectrum of a protein's role.

1) Graph Neural Networks (GNN) Technique

Gligorijević et al. [55] introduced DeepFRI, a Graph Convolutional Network that improves protein function prediction by utilizing both a protein language model and structural data, visualizing the interactions of amino acids in 3D structures. Li et al. [56] developed DeepPFP-CO using the GCN to optimize protein function prediction through co-occurrence patterns of GO terms.

Li et al. [57] introduced DeepPFP-CO, a deep learning framework for protein function prediction using GO term co-occurrences. It comprises two main parts: a feature combination module that extracts and merges sequence and subsequence-based and PPI network data, and a function prediction module that employs a GCN to enhance accuracy by utilizing a correlation matrix derived from GO term co-occurrences.

Taha et al. [58] introduced iPFPi, a classifier system designed to predict functions for un-annotated proteins. iPFPi associates un-annotated protein *P* with GO annotation terms that share semantic similarity with *P*. Both *P* and a GO annotation term *T* are represented by their respective characteristics, where *P*'s characteristics include GO terms found in the abstracts of biomedical literature associated with *P*, and *T*'s characteristics include GO terms found in the abstracts of biomedical literature associated with proteins annotated with function *T*.

Wang et al. [59] launched DeepBIO, a deep learning platform featuring various methods such as GNNs and convolutional networks for predicting biological sequences and functions, with enhanced visualization tools for model analysis. Ioannidis et al. [60] proposed a deep learning framework using coefficients in multi-relational protein-to-protein networks to forecast protein functions, incorporating a graph neural network technique. Abdine et al. [61]'s Prot2Text system is a multimodal approach combining GNNs and ESM, using GPT2 for textual descriptions of protein functions.

2) Graph Attention Network (GAT) Technique

Lai et al. [62] introduced GAT-GO, a method based on the graph attention network (GAT) framework. This approach enhances protein function prediction by incorporating predicted structural data and protein sequence embedding. RaptorX is utilized within this method to predict a protein's structural details. Moreover, the methodology employs a CNN-based sequence feature encoder that accepts three distinct sequence attributes. These are transformed into residue-level feature vectors. Specifically, the three input characteristics consist of the one hot encoded primary sequences, the sequence profile, and the residue-level sequence embedding sourced from a protein-centric language model.

Mostafavi et al. [63] introduced a novel technique known as GeneMANIA, designed for the rapid integration of diverse input data sources to predict protein functions, making it suitable for deployment on a web server. GeneMANIA employs a label propagation algorithm, which assigns discriminant values by optimizing a cost function. This cost function penalizes discrepancies in discriminant values between neighboring nodes within the network and deviations between the discriminant values of nodes and their label biases.

Peña-Castillo et al. [64] curated a uniform compilation of functional genomic data for mice. This dataset served as the foundation for training independent classifiers and generating function predictions, defined by GO terms, for a total of 21,603 mouse genes. The best predictions from multiple submissions were combined and analyzed to assess the strengths and weaknesses of current functional genomic datasets and to evaluate the performance of function prediction algorithms.

Li et al. [65] developed DeepGATGO, a protein function prediction method combining Graph Attention Networks (GATs) and contrastive learning, focusing on optimizing embeddings from sequence and GO label data and emphasizing structural and label dataset attributes. Baranwal et al. [66] introduced Struct2Graph, focusing on predicting PPIs and functions from the structural data of protein globules using a graph attention network, emphasizing protein-protein complex formations derived from data via a multi-layered GCN.

D. Convolutional-Based Sub-Category

Convolutional-based Networks have found applications in predicting protein function by leveraging their strength in detecting local patterns and hierarchical features. Originally designed for image processing tasks, they have been adapted to handle one-dimensional sequences like amino acid chains or two-dimensional representations like contact maps in protein structures. Convolutional Layers are the core building blocks of CNNs. They scan the input through filters, identifying local features like motifs or domains in proteins that are crucial for function. Pooling Layers reduce the dimensionality of the feature maps generated by the convolutional layers, retaining the most salient information and making the network more computationally efficiency.

1) Convolutional Neural Networks (CNN) Technique

Kulmanov and Hoehndorf [15] proposed a new technique for predicting protein functions using only their sequences. This approach blends a deep convolutional neural network (CNN) with predictions based on sequence similarity. The CNN model identifies motifs in the sequence indicative of protein functions and integrates this information with the functions of proteins that have similar sequences.

Kulmano et al. [13] proposed a method using multi-layered neural networks to predict protein functions from sequences and interactions. It extracts features from sequences and represents proteins based on their network position. These features are integrated into a deep neuro-symbolic model reflecting the GO's structure, aiming to improve function prediction across the ontology hierarchy.

Golkov et al. [14] introduced a deep learning approach designed to infer the biological function of molecules directly from their unprocessed 3D approximated electron density and electrostatic potential fields. The method determines protein function using EC numbers, derived from the approximated electron density field.

Kumar et al. [16] unveiled a streamlined architecture using dilated-CNN for proficiently modeling protein sequences and predicting their functions. The dilated-CNN is adept at capturing a broad spectrum of interactions among amino-acid k-mers, encompassing proximal and distant interactions. A lower dilation rate zeroes in on neighboring protein segments, while a higher rate targets the more remote protein segments.

Giri et al. [67] introduced a sophisticated multi-source multi-modal framework capable of precisely predicting protein functions. This system employs protein interaction data along with two modalities for prediction: the base amino acid sequences and the 3D structures from the protein data bank. A standout feature of their approach is the transformation of 3D PDB structures into 2D voxels using ResNet-50.

Cai et al. [68] put forth an advanced deep-learning classification system, dubbed SDN2GO, for protein function prediction. This system employs convolutional neural networks to discern and pull features from sequences, protein domains, and established PPI networks. A weight classifier merges these features, enabling precise predictions of GO terms.

III. Statistical Inference Category

A. Probabilistic Analysis Sub-Category

Probabilistic analysis methods in machine learning offer a different approach to predicting protein function by focusing on the uncertainty and probabilistic nature of biological data.

1) Naïve Bayes-Based Technique

You et al. [69] introduced NetGO, an online platform designed for protein function prediction. NetGO enhances the efficiency of large-scale Automated Function Prediction (AFP) for proteins by integrating extensive protein-protein network data. The foundational concept behind NetGO is the amalgamation of six distinct methods within the LTR framework to boost AFP outcomes. Among the five components are Naïve Bayes, BLAST-KNN, LR-3mer, LRInterpro, and LR-ProFET, all of which are derived from GOLabeler that utilizes protein sequence data.

Silla and Freitas [70] introduced a novel global-model technique for predicting protein functions using hierarchical classification. In this method, a singular global classification model is constructed by accounting for all classes within the hierarchy. This approach is an enhancement of the naive Bayes flat classification algorithm. The term "global" in the classification model denotes its inclusivity of all hierarchical classes, in contrast to the typical approach of constructing multiple local classification models.

Tang [71] employed the naïve Bayes classifier tool from the FEATURE software package to pinpoint protein functions. This classifier discerns and amalgamates distinguishing characteristics into a naïve Bayes model, enabling differentiation between functional and non-functional sites. The model acts as a binary classifier, categorizing sites as either positive (having the function) or negative (lacking the function). This classifier is trained with FEATURE vectors related to sites that either have or lack the said function.

B. Structural-Based Sub-Category

1) Decision Tree (DT) Technique

Singh and colleagues [72] introduced a novel decision tree induction approach for determining protein functions. This method employs an uncertainty measure for optimal attribute selection. Their proposed technique prioritizes packages of SDFs (Sequence Derived Features), allowing for a deeper exploration in creating the decision tree, instead of merely excluding them. Consequently, the model produces a decision tree with enhanced depth. A deeper tree ensures that more tests are conducted prior to assigning a functional class, leading to predictions that are more precise compared to current methods.

Yedida et al. [73] designed a decision tree-based system for automatic protein function labeling. This system detects proteins with similar characteristics using an innovative method to measure the similarity between two protein sequences. Proteins' biological roles are characterized using their GO annotations. The decision trees within the system are formulated based on the GO annotations of similar proteins and the extent of their sequence resemblance.

Deen and Gyanchandani [74] introduced a prediction model that employs decision tree classifiers, including Information Gain, Gini-Index, and Random Forest. They evaluated the performance of these classifiers based on the achieved accuracy. The machine learning classifiers are constructed using features from the membrane cell sequence, with some critical functions relying on PseAAC (Pseudo Amino Acid Composition) descriptors.

Pan [75] utilized a blend of Decision Tree and Support Vector Machine techniques for protein prediction by extracting rules. The fundamental principle of this approach is that effective interpretation can guide biological experiments and potentially enhance prediction accuracy.

2) Support Vector Machine (SVM) Technique

CZ et al. [76] introduced a web-based tool, SVMProt, designed for SVM-based classification of proteins into functional families based on their primary sequences. The SVMProt system is trained using representative proteins from various functional families as well as seed proteins from the curated Pfam protein families. It gathers distinct functional protein classes from multiple databases, encompassing major protein categories such as enzymes, receptors, transporters, channels, and proteins that bind to DNA or RNA. At the heart of SVMProt is the SVM program.

Barot [77] introduced a method called deepNF, designed to detect topological patterns spanning various PPI networks for the purpose of identifying protein functions. The method allocates distinct layers early in the deep autoencoder process. DeepNF, in its final stage, uses these features to train an SVM to predict the specific function of each protein.

Deen and Gyanchandani [78] explored protein function prediction utilizing kernel methods, improved the support vector classifier, and boosted classification accuracy. They discovered that the classification accuracy for functional protein classes using the RBF kernel is 97.27%.

Saha and Shill [79] introduced a technique that combines support vector machines and fuzzy logic to predict protein secondary structure and function without needing alignment. This method determines the optimal hyperplane for the support vector machine using membership values.

Jung et al. [80] utilized the SVM algorithm for protein function identification. For every GO term, they built a dataset that includes proteins labeled with that GO term and those without that annotation. Given the significant imbalance in the datasets, they undertook undersampling of the negative class. This ensures that the number of proteins in the training dataset is equal between the positive and negative classes.

Yadav et al. [81] introduced a supervised strategy for predicting the main functional classes and subclasses of proteins. They employed the SVM to develop a three-tier model with an optimally selected number of features. The first level determines whether a protein is an enzyme or non-enzyme. The second level classifies the functional category of the enzyme, while the third level specifies the subfunctional class.

IV. Ensemble Learning Category

Ensemble learning is a technique where multiple models are trained to solve the same problem and their predictions are combined to get a final result. It improves the performance of by reducing overfitting, increasing stability, and boosting the predictive power of the model. In the context of predicting protein function, ensemble learning can be particularly useful due to the complexity and variability of biological data.

A. Extreme Gradient Boosting (XGBoost) Technique

Gou et al. [82] developed four individual and 11 combined models to investigate plant protein functions using gathered experimental data. The research team utilized techniques such as XGBoost, random forest (RF), SVM, and feedforward neural network (FFNN). Out of all the models tested, XGBoost emerged as the top performer and was chosen as the primary algorithm for the suggested approach.

Wang et al. [83] introduced a technique using deep learning combined with XGBoost, termed DeepPPISPXGB, to forecast protein-protein interaction sites and protein functionality. The deep learning framework acted as a mechanism to filter out superfluous data from protein sequences. The Extreme Gradient Boosting method was employed to build a classifier for predicting the PPI sites and their functions.

Kool [84] introduced a technique that predicts residues of catalytic active sites, relying on the eXtreme Gradient Boosting (XGBoost) tree-based classification strategy. The model's performance on benchmark sets for catalytic active sites matched that of other contemporary methods, albeit with considerably fewer features.

B. Random Forest Technique

Srivastava et al. [85] evaluated the performance of two data-mining methods, Random Forest and SVM, in predicting protein function. For the Random Forest method, they utilized 7 features for their predictions.

Okada et al. [86] presented a modified version of Random Forest aimed at enhancing the accuracy of predictions related to protein function based on structural information. They incorporated microenvironment descriptors from the FEATURE framework. The authors tested their system using a balanced collection of seven function models sourced from various public databases. They systematically assessed the accuracy of their Random Forest-based system in comparison to SVM and NB. The findings revealed that, with meticulous parameter adjustments, Random Forest's accuracy surpassed that of NB and was on par with SVM.

Hakala et al. [87] crafted an ensemble system that integrates GO predictions from both random forest (RF) and neural network (NN) classifiers to identify protein functions. The RF and NN models utilize features obtained from BLAST sequence alignments, taxonomy, and protein signature analysis tools. The researchers also detailed experiments involving a NN model that exclusively processes the amino acid sequence as its primary input through a convolutional layer.

V. Neighborhood-Based Category

Unsupervised learning in the context of predicting protein function is a computational approach that does not require labeled data for training. In other words, the algorithm learns the inherent structure or patterns in the data without using any predefined classifications. Unsupervised learning methods in predicting protein function are often used in exploratory phases of research or when labeled data is scarce. They are valuable tools but serve as a complementary approach to supervised and semi-supervised methods, which provide more accurate predictions when labeled data are available. Fig. 2 depicts the processing of K-Nearest Neighbor (KNN) Technique, which is one of the unsupervised learning techniques.

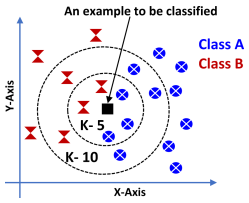


Fig. 2: Depicting the processing of KNN Technique

A. Autoencoder Technique

Gligorijević et al. [88] introduced deepNF, a technique for predicting protein function through network fusion. Utilizing Multimodal Deep Autoencoders, the method distills high-level protein attributes from various disparate interaction networks. It amalgamates STRING networks to create a unified low-dimensional representation rich in high-level protein characteristics. In the early phases of the multimodal autoencoder, distinct layers are employed for different types of networks. These layers are eventually unified into a single bottleneck layer, from which features are extracted for the purpose of protein function prediction.

Miranda and Hu [89] introduced a deep learning framework that relies on a stacked denoising autoencoder for the purpose of predicting protein function. This system is designed to harvest robust features that enhance the accuracy of predictions. These extracted features are subsequently input into a multilabel SVM for the classification task.

Miranda and Hu [90] introduced an autoencoder framework designed for predicting protein functions. This model is geared towards isolating important features for insightful representation. Using a mechanism called mutual competition, the hidden layers employ a winner-take-all strategy to engage neurons in a competitive process for encoding pertinent features. The approach comprises two main components: (1) a winner-take-all operation that identifies a subset of key hidden neurons, and (2) a subsequent step that incentivizes the activated neurons in the winning group while penalizing those in the losing group.

Bonetta and Valentino [91] conducted a review of machine learning methods applied to protein function prediction. They highlighted the evolution of feature types utilized by these algorithms, ranging from traditional physicochemical characteristics and amino acid makeup to more contemporary features extracted from biomedical texts. Additionally, they discussed the incorporation of autoencoder-generated feature representations, as well as techniques for feature selection and reducing data dimensionality. Dhanuka et al. [92] introduced a semi-supervised, autoencoder-driven deep learning approach specifically for protein function prediction. In this method, a collection of autoencoders is trained semi-supervisedly using protein sequences. Importantly, each autoencoder is dedicated to predicting just one specific protein function.

B. K-Nearest Neighbor (KNN) Technique

Törönen and Holm [24] formulated PANNZER, a classifier based on weighted K-nearest neighbor principles, which aims to predict protein functions. The researchers not only showcased the application of PANNZER but also introduced supplementary functionalities such as taxonomic filtering and gene name prediction. Lan et al. [23] presented the Multi-Source k-Nearest Neighbor (MS-kNN) technique designed for function prediction. This algorithm identifies the k-nearest neighbors of a given query protein using diverse similarity metrics and forecasts the protein's function through a weighted average of its neighboring proteins' functions. The study employed three distinct data sources to compute similarity scores: sequence homology, protein-protein interactions, and gene expression patterns.

Re et al. [93] developed a transductive gene ranking approach that uses kernelized score functions to exploit the topology of biomolecular networks, capturing functional gene connections. This method, which extends average, nearest neighbor, and k-nearest neighbor distances from "positive" genes of a specific function, employs a flexible kernel to represent gene functional similarity. The approach is adaptable, allowing for different local score functions and kernels to create various network-based gene function prediction algorithms.

Yh et al. [25] presented a new feature for conducting BLAST sequence alignment, which allows users to search for proteins within the functional families predicted by SVM-Prot that are similar to a query protein. This aids in the focused study of specific proteins from the SVM-Prot predicted families that may be relevant to the query protein. To further improve the functional prediction capabilities, this version of SVM-Prot incorporates two additional machine learning techniques—K nearest neighbor (kNN) and probabilistic neural networks (PNN)—to enable a more comprehensive evaluation of protein functional families.

Yw et al. [94] introduced GODoc, a robust framework for predicting protein functions using Gene Ontology (GO). The team developed three innovative voting mechanisms based on the kNN algorithm, incorporating a training phase to address the challenge of multi-label prediction, especially given the disparity between the number of GO terms and localization sites. To identify suitable candidate proteins, the authors employed the kNN algorithm. The GO terms predicted for the target protein are then determined through a voting process involving the GO terms of these candidate proteins. A weighted voting approach is used, where candidates more similar to the target protein carry greater weight in the voting process.

VI. Attention Mechanisms Category

Attention mechanisms in machine learning have proven effective in predicting protein function by focusing on specific, relevant parts of protein sequences or structures. Attention mechanisms allow the model to focus on specific parts of the protein sequence that are more relevant for a particular task. In essence, they weight different parts of the input differently, depending on the context within which they appear. This is particularly useful for sequences of varying lengths and complexities, as seen in proteins. After passing through the attention layers, the model outputs predictions regarding the protein's function, often in the form of probability scores for belonging to various functional categories. Providing high-dimensional embeddings and contextual awareness, these mechanisms offer several advantages: (1) They capture distant relationships in protein sequences that are crucial for functional determination, (2) Being highly parallelizable, they are efficient on large datasets, and (3) Often outperforming traditional methods, they provide state-of-the-art results.

A. Multi-Attention Mechanism Technique

Zheng and colleagues [95] utilized the three-dimensional structures predicted by AlphaFold, along with additional non-structural indicators, to create PredGO, a comprehensive method for annotating Gene Ontology (GO) functions of proteins. The team employed a pre-trained language model, geometric vector perceptrons, and multi-attention systems to glean diverse features of proteins, which were then integrated for the purpose of predicting their functions.

Li et al. [96] presented a deep ensemble approach for identifying protein functions. This technique employs an amalgamation of various protein encodings, including local descriptors, auto-covariance, conjoint triads, and pseudo amino acid composition. The ensemble model comprises several components like input, convolution, and an attention mechanism that utilizes multi-attention systems to capture critical features. Ranjan et al. [97] introduced a Multi-Attention Mechanism aimed at determining the optimal number of significant n-mers within a protein sub-sequence. Rather than relying solely on a single layer of attention mechanisms, their proposed approach utilizes a series of parallel attention layers to assess the importance of protein-related words or segments. These attention distributions from each layer are aggregated to generate an optimized distribution of attention scores.

Wang et al. [98] introduced a comprehensive deep neural model, termed MMSMA, designed for predicting protein function. Initially, MMSMA pulls in diverse feature sets from protein sequences, such as one-hot encoding attributes, evolutionary metrics, deep semantic markers, and physiochemical overlapping properties. Following this, a specialized deep neural network is constructed for each feature view to facilitate in-depth feature extraction and initial classification. MSMA can identify both localized patterns and long-range dependencies within protein sequences. An adaptive decision-making mechanism that leverages multiple views is employed to consolidate the individual classification outcomes.

VII. Multi-Task and Multi-Label Techniques and Protein Language Models for Detecting Protein Functions

A. Multi-task and Multi-Label Techniques

1) Deep Neural Networks (DNN) Technique

The DNN method for detecting protein functions employs multi-task and multi-label approaches, processing extensive protein data (sequences, structures, annotations) for neural network training, including sequence encoding and normalization [99]. Central to this technique is multi-label classification, where DNNs predict various protein functions, outputting probabilities for each. This process requires a unique architecture and a specialized loss function to accurately handle the complexity of the task. Additionally, the technique incorporates multi-task learning, which significantly enhances its generalization and accuracy. By training on related tasks, it allows for robust predictions, particularly in understanding diverse protein functions, demonstrating the sophisticated capabilities of this approach in computational biology.

2) Multi-Layer Perceptron (MLP) Technique:

MLPs for multi-label protein function prediction can predict multiple functions per protein in one pass, using specialized loss functions like modified cross-entropy. They feature shared hidden layers and task-specific output layers for related functions. Training uses data augmentation or transfer learning, making their architecture and training distinct for multi-task and multi-label prediction, unlike non-neural network methods.

3) Recurrent Neural Networks (RNN) Technique

RNNs excel in predicting multi-task and multi-label protein functions, uniquely capable of handling proteins' complex functionalities. They stand out by producing multiple predictions per sequence, reflecting proteins' multifaceted roles. Additionally, combining RNNs with other networks like CNNs enhances prediction accuracy, leveraging CNNs for feature extraction and RNNs for sequential processing, leading to more effective and precise function predictions.

4) Long Short-Term Memory (LSTM) Technique

In multi-task learning, especially for complex areas like protein function prediction, LSTMs are highly effective due to their ability to process sequential data and handle multiple tasks and labels simultaneously. They excel by creating shared representations for related tasks, improving accuracy. Their gated mechanism allows for selective information retention, essential in understanding the diverse biological roles of proteins. LSTMs, when integrated with CNNs or attention mechanisms, improve in predicting complex protein functions.

5) Graph Neural Networks (GNN) Technique

GNNs use graph structures to identify multi-task and multi-label protein functions, differing from other methods. These graphs, with nodes as proteins and edges as interactions, reflect natural biological networks, capturing complex relationships

better than vector-based methods. GNNs apply convolutional layers to these graphs, aggregating information from neighboring proteins, crucial for understanding protein functions. Their structure allows learning at various levels, suitable for proteins' diverse functions and biological roles.

6) Graph Attention Network (GAT) Technique

GAT excels in analyzing protein interaction graphs by weighting protein interactions variably, aiding in multi-label contexts where proteins have multiple roles. Representing proteins as distinct nodes, GAT's attention mechanism zeroes in on crucial neighboring features for precise function prediction. This method excels in complex multi-label classification and multi-task learning, enabling varied protein function prediction and improved generalization.

7) Convolutional Neural Networks (CNN) Technique

CNNs are highly effective in protein function detection, offering multi-task and multi-label prediction through multiple output layers. Their design, including normalization and dropout layers, prevents overfitting and enhances generalization. CNNs automatically extract relevant features, outperforming methods needing manual selection. Integrating CNNs with RNNs or attention mechanisms further boosts accuracy by capturing local and global dependencies.

8) Naïve Bayes-Based Technique

In multi-label protein function prediction, the Naïve Bayes method independently assesses each function, calculating its presence probability based on observed features. This enables simultaneous predictions for multiple functions per protein. Its simplicity and computational efficiency make it ideal for quick, initial analyses in large datasets.

9) Support Vector Machine (SVM) Technique

In multi-label classification, SVM uses one-vs-rest (OvR) or one-vs-one (OvO) methods, training multiple classifiers for distinguishing classes. For multi-task learning, SVM uses shared information across tasks, often through joint feature selection or regularization, improving protein function prediction. Also, SVM excels in high-dimensional spaces, common in biology, using the kernel trick to efficiently operate in high-dimensional feature space without explicit mapping.

10) Random Forest Technique

Random Forest efficiently handles multi-label data, naturally predicting multiple labels without needing special adaptations. It aggregates predictions from each decision tree, which uses a random subset of features. This approach suits protein function prediction, as it explores various feature combinations and reduces overfitting in high-dimensional data.

11) Autoencoder Technique

The autoencoder compresses protein data, retaining key features for multi-label classification. The decoder then reconstructs this data, imperfectly, enabling the model to learn essential protein information for multi-task learning.

12) Multi-Attention Technique

The core of this technique involves multiple attention layers designed to focus on distinct aspects of protein data. This differs from traditional multi-task and multi-label methods. The Multi-Attention Mechanism identifies protein functions and understands their interdependencies. It breaks down protein structures into smaller segments for detailed analysis, effectively handling complexities in protein structures.

13) Decision Tree (DT) Technique

DT techniques uniquely detect multi-task and multi-label protein functions by optimizing multi-label assignments during tree construction, inherently considering multi-label aspects in decision-making at each node, unlike other methods.

B. Protein Language Models

Protein language models for protein function prediction represent a fascinating intersection of bioinformatics and artificial intelligence [100-103]. These models use techniques similar to those in natural language processing (NLP) to understand and predict the functions of proteins based on their amino acid sequences.

In a way, amino acid sequences in proteins are akin to words in a language. Each sequence has its unique 'meaning' or function, just like words in a sentence. Protein language models leverage this similarity by using algorithms similar to those used in NLP to interpret these sequences. Here's how they work:

1. **Sequence Analysis**: Just as language models analyze word patterns, protein models analyze amino acid sequences. They look for patterns that are often associated with specific functions or structures.
2. **Training on Databases**: These models are trained on vast databases of known protein sequences and their functions. This training allows the models to learn the correlation between sequence patterns and their corresponding functions.
3. **Predicting Protein Function**: After training, these models can predict the function of a new, unknown protein sequence. They do this by comparing the new sequence to the patterns they've learned.
4. **Deep Learning Techniques**: Advanced models use deep learning techniques, such as convolutional neural networks (CNNs) or recurrent neural networks (RNNs), to improve their prediction accuracy. These techniques can handle the complexity and variability in protein sequences.
5. **Applications**: This technology has vast applications in drug discovery, understanding disease mechanisms, and synthetic biology. By predicting protein functions, scientists can identify potential targets for new drugs or understand how genetic mutations might lead to diseases.

The development of protein language models is an ongoing field of research, continually evolving with advancements in AI and a deeper understanding of biological processes.

VIII. COMPARATIVE EVALUATIONS

In this section, we explore the techniques outlined in this paper, assessing them based on four fundamental criteria: their foundational principle, the rationale behind their application, the critical conditions for optimal performance, and any limitations. Table 1 highlights the techniques that employ machine learning to predict protein functions. Our goal is to provide a comprehensive understanding of each technique's strengths and weaknesses, and to determine their appropriateness for specific tasks.

Table 1. Evaluating each machine learning technique for identifying protein function in terms of the following four criteria: its underlying principle, its justification, its conditions for optimal performance, and its limitations

Tech.	Papers	Technique Essential Concept	Rationale Behind the Usage of the Technique	Conditions for the Optimal Performance of the Technique	Limitations of the Technique
Deep Neural Networks (DNN)	[40-44]	DNNs utilize the multiple layers of the network to learn complex, hierarchical features from encoded protein data. The input data are converted into a numerical format that the DNN can understand. A technique like one-hot encoding can be used. DNNs have multiple interconnected layers, including input, hidden, and output layers, allowing them to learn complex features. The network aims to minimize a loss function, often something like categorical cross-entropy, to adjust the model weights appropriately. Methods like saliency maps can be used to understand which regions of the protein were crucial for making a particular prediction, providing some level of insight into what the DNN has learned	(1) DNNs are effective in analyzing complex protein data like sequences and 3D models, excelling in feature extraction and pattern recognition for protein functions, (2) They leverage large protein databases, thriving in data-rich environments, (3) With ample data and resources, DNNs can decipher intricate relationships, making them ideal for studying the links between protein structures and functions, (4) DNNs are versatile in processing diverse data, enabling a comprehensive approach to predicting protein functions, (5) Fine-tuning of pre-trained models is efficient, aiding knowledge transfer and performance on unfamiliar datasets, (6) DNNs automate feature extraction, surpassing the limitations of traditional, manually-engineered methods	(1) The architecture of the neural network should be designed to be complex enough to recognize intricate patterns, but not so elaborate that it risks overfitting, (2) To prevent overfitting, methods such as dropout and [1]L1/L2 regularization should be implemented, (3) Activation functions like ReLU[2], Tanh, and Sigmoid are generally utilized, (4) The learning rate should be configured to allow for rapid convergence while avoiding overshooting the ideal solution, (5) The batch size needs to be fine-tuned to strike a balance between computational efficiency and the stability of the training process, (6) Depending on the specific needs of the problem, optimization algorithms such as Adam, RMSprop, or SGD should be selected.	DNNs have limitations in detecting protein function, primarily due to scarce and imbalanced training data, which affects their generalizability and can lead to biased results. The complexity of biological systems makes it hard for these models to capture all relevant factors. They are also resource-intensive and struggle with feature selection, further challenging their efficacy. The lack of interpretability of these "black box" models is a key concern in the biomedical field. Data inaccuracies due to experimental errors or incomplete annotations hinder performance. Also, the multi-functionality of proteins and evolutionary changes complicate categorization tasks, and integrating other biological data to enhance predictions remains challenging.
Multi-Layer Perception (MLP)	[45-47]	Each neuron in a hidden layer takes a weighted sum of the outputs from the previous layer, applies an activation function, and passes its output to the next layer. The final layer in an MLP typically uses an activation function suitable for the task at hand (e.g., softmax for multi-class classification). The output could represent the probability that the input protein performs a particular function or belongs to a specific class of proteins. Functions like ReLU (Rectified Linear Unit), sigmoid, or tanh are commonly used to introduce nonlinearity into the network. During training, the MLP adjusts its internal parameters (weights and biases) using an optimization algorithm, commonly some form of gradient descent. Techniques like dropout, L1/L2 regularization, or early stopping can be used to mitigate this.	(1) The link between a protein's structure and function is non-linear. MLPs, using activation functions like ReLU, are effective at capturing these complexities, (2) MLPs can autonomously identify crucial features during their training, which is a boon given the complexity of protein structures, as it lessens the need for manual feature picking, (3) While not as easily interpretable as some models, MLPs' hidden layers do offer valuable insights into significant features for predictions, which is vital in biological research for new discoveries, (4) MLPs integrate well with other techniques like sequence alignment to provide a more complete solution for determining protein functions, and (5) The structure of MLPs can be modified with multiple layers and neurons, making them adaptable for different types of data and feature complexities.	(1) Ensuring a balanced dataset for training the MLP is crucial, as an imbalance in classes could distort performance measurements, (2) A network with too few layers may fail to capture intricate relationships in the data, while an overly deep network risks overfitting, (3) The optimal number of neurons per layer should be tailored to match data's complexity, (4) Activation functions (e.g., ReLU, Tanh) can influence how quickly the model trains and reaches convergence, (5) Setting a learning rate that is either too high may lead to oscillations, while a rate that is too low can result in sluggish convergence, (6) The iteration count should be sufficient for the model to converge but not so excessive that it leads to overfitting, (7) Overfitting can be mitigated through techniques like dropout, (8) Model's performance can vary depending on how protein sequences are represented (e.g., sequence features, structural attributes).	(1) MLPs ignore proteins' spatial aspects, (2) Large, labeled datasets are scarce but essential for MLPs in protein work, (3) MLPs may require detailed input features like amino acid properties, (4) Learning can get stuck, yielding suboptimal results, (5) Complexity and noisy data can lead to overfitting in MLPs, (6) MLPs' lack of transparency is a hurdle in bioinformatics, (7) Hyperparameter Focus (Tuning is crucial for performance), (8) MLPs can demand high computational resources, (9) MLPs can't handle time-varying protein features well, (10) MLPs struggle with poorly represented protein functions, (11) Complex output spaces may be needed for diverse protein functions, (12) MLPs can't model biological feedback loops, (13) MLPs find multi-functional proteins in varying contexts hard to capture, (14) MLPs overlook post-translational modifications in proteins.

[1] L1/L2 regularization prevent overfitting in ML by adding penalties based on the model coefficients' absolute values (L1) or squares (L2) to the loss function.
[2] ReLU, or Rectified Linear Unit, is a function used in neural networks defined as f(x) = max (0, x), which returns x if x is positive and 0 otherwise.

Recurrent Neural Networks (RNN) [18-29, 48, 49]	By utilizing hidden states, RNNs possess a kind of 'memory,' which enables them to identify intricate relationships between the amino acids in a sequence. This makes RNNs highly valuable for applications such as predicting protein functions. To capture long-range dependencies within protein sequences more effectively, specialized variants of RNNs like LSTM and Gated Recurrent Units (GRU) are frequently used. The training process for RNNs employs a customized form of the backpropagation algorithm, which allows the network to learn from its errors and fine-tune its weights. Techniques like LSTM and GRU are often used to counteract the issues of vanishing or exploding gradients.	RNNs use hidden states to remember past sequence elements, effectively capturing context like how protein segments are influenced by their surrounding sequence. RNNs use consistent parameters, aiding generalization and reducing overfitting, which is valuable in limited protein data. RNNs process sequences quickly, making them efficient for large datasets. Advanced RNNs like GRUs excel at learning complex patterns, crucial for understanding non-linear interactions in proteins. RNNs facilitate end-to-end models that translate raw sequences into functional predictions, bypassing extensive feature engineering. Their architecture allows easy integration with other neural network types to incorporate extra data, like protein interactions, improving function predictions.	(1) Ensemble Methods: Using multiple RNNs and then averaging or voting their outputs can lead to better results. (2) Adding randomness by zeroing some input/output units during training updates can reduce overfitting risk. (3) Batch Norm: This technique improves training stability by normalizing network activations during learning. (4) Early Stop: Monitoring validation loss and stopping training when it increases can prevent overfitting. (5) Init Strategies: Choice of weight initialization like Xavier or He methods greatly affects model performance. (6) Methods like learning rate annealing help control learning rate during training. (7) Networks with more layers and units can capture complex relationships but may overfit. (8) Using attention mechanisms helps networks focus on crucial subsequences for classification.	(1) RNNs struggle with distant interactions due to gradient issues, limiting complex motif capture in protein sequences, (2) RNN training is costly, (3) Complex RNNs tend to overfit small, less diverse datasets, (4) RNNs' opacity hinders clear understanding of protein function, (5) Functionally similar proteins can differ structurally, posing accuracy challenges, (6) Alignment quality impacts RNN performance, due to alignment's complexity, (7) RNNs struggle with diverse protein sizes, computationally demanding, (8) Laborious, time-consuming labeling hampers accurate function prediction, (9) Unknown protein functions and limited classes lead to training data imbalance, (10) RNNs excel at sequences but need complex designs for diverse data integration, (11) Evolving proteins and multiple functions challenge static models' generalization.
Long Short-Term Memory (LSTM) [50-53]	LSTMs evaluate amino acid sequences to uncover intricate patterns that are indicative of protein functions. Composed of a sequence of interconnected gates—such as the forget gate, input gate, and output gate—and designated cell states, LSTMs are engineered to determine what data to preserve and what to overlook. This architecture aids the network in maintaining a continuous focus on relevant aspects throughout the entire amino acid sequence, thus enhancing its capability for accurate predictions, even when dealing with lengthy sequences. While it's occasionally feasible to explore LSTM layers for significant sequence features in function prediction, this is generally tougher than using standard bioinformatics methods	(1) Protein function relies on amino acid arrangement; LSTMs excel in learning sequences, suiting this purpose. (2) Proteins contain many amino acids, posing issues for traditional RNNs due to sequence length. LSTMs handle long-range dependencies and prevent gradient problems. (3) Proteins have diverse features like motifs and structures. LSTMs process complex data, enabling thorough sequence analysis. (4) LSTMs detect sequential patterns, aiding in identifying functional elements within protein sequences. (5) Well-trained LSTMs generalize effectively, robustly detecting protein function across species or conditions. (6) Despite complexity, LSTMs can be interpreted using techniques, revealing insights into protein function via understanding significant model features.	(1)Maintaining similar ranges for feature values can enhance training stability and speed, (2) Networks that are too shallow may not capture protein function complexity, and excessively deep networks risk overfitting, (3) Incorporating attention mechanisms helps the model focus on crucial segments of protein sequences that hold more informative value for specific functions, (4) Selecting an appropriate learning rate is crucial for getting convergence, (5) Larger batches provide more stable convergence but might be suboptimal, whereas smaller batches introduce noise but can reach a superior local minimum, (6) [3]TBPTT efficiently handles extensive sequences, (7) Pre-trained models can be fine-tuned for the specific task of protein function detection, (8) Aggregating predictions from multiple LSTM models enhances performance.	1) LSTMs might struggle with diverse protein sequences compared to their training set. (2) LSTMs need abundant, well-annotated data for training, which could be lacking for certain protein families/functions. (3) LSTMs are sensitive to input length, problematic for proteins spanning a wide amino acid range. (4) LSTMs are often seen as opaque models, hindering interpretation, notably in scientific use. (5) Determining optimal features (e.g., amino acids, secondary structure) for LSTMs is complex. (6) Imbalanced protein function data can lead to biased predictions in many datasets. (7) Some proteins have multiple functions, and LSTMs might not be well-suited to capture this multifunctionality without specific architectural or training set adjustments.
Graph Neural Networks (GNN) [54-58]	Nodes and edges possess attributes that encompass initial biochemical characteristics like amino acid composition and charge. These attributes are translated into feature vectors. Through an iterative process, GNNs disseminate information across the graph, updating attributes associated with nodes and edges. This enables the model to comprehend local and global structural attributes. GNNs employ aggregation functions (e.g., sum and average) to condense the attributes of neighboring nodes to revise individual node attributes. A readout function amalgamates nodes' attributes, generating a feature vector representative of the entire graph. This serves the final objective, such as the classification of a protein's function. GNNs elucidate residues or interactions pivotal for specific protein functionalities via the learning process.	(1) Proteins can be naturally represented as graphs, where nodes represent amino acids or secondary structure elements, and edges represent spatial or functional interactions, (2) GNNs are capable of capturing both local (e.g., motifs, patterns) and global (e.g., overall topology) features of the graph, which are crucial for understanding protein function, (3) GNNs are computationally efficient when dealing with large-scale graphs, as they perform localized computations, (4) GNNs are robust to different graph sizes and can incorporate new nodes or edges without the need for re-training, making them scalable, (5) GNNs can handle heterogeneous data, including various types of nodes and edges (e.g., hydrophobic interactions, hydrogen bonds), and can also incorporate node and edge attributes, (6) GNNs can include attention mechanisms to highlight important nodes and edges.	(1) The utilization of functions such as ReLU, leaky ReLU, or tanh can influence the model's performance. (2) The performance of the model can be impacted by the approach chosen for passing messages between nodes (e.g., sum, mean, LSTM). (3) The outcomes can also be influenced by the method employed to derive a graph-level representation from node-level features. (4) Optimal solutions might be overlooked with a high learning rate, whereas a low rate could lead to gradual model convergence. (5) The learning process's speed and stability can both be influenced by the chosen batch size. (6) Employing early stopping or other criteria is advisable to determine the point at which training has adequately converged, (7) The number of layers should be carefully chosen; too few may lack expressiveness, while too many can lead to overfitting or vanishing gradient	(1) GNNs demand substantial computation for large protein data, (2) Biological data is sparse and noisy. GNNs handle sparse data but may suffer performance, (3) Performance depends on accurate graph representation of biological structure. Errors in graph formation yield misleading results, (4) Protein function is influenced by diverse data (genomic, proteomic), GNNs emphasize graph data, neglecting other features, (5) GNNs are hard to interpret, (6) Complex GNNs with high-dimensional biological data risk overfitting, limiting generalization, (7) Inadequate or missing node features hinder GNN performance, (8) Many GNNs can't manage scalability in large biological networks, (9) GNNs struggle to capture multiscale features of diverse biological scales, (10) Due to biological complexity, GNN predictions' validation is tough and time-consuming.

[3] TBPTT "Truncated Backpropagation Through Time" is a variant of the backpropagation algorithm used for training certain types of neural networks, particularly RNNs. In standard backpropagation, the algorithm calculates gradients of the loss function with respect to the weights of the network, used for updating the weights.

Graph Attention Network (GAT) [59-61]	The key novelty of GAT stems from its utilization of an attention mechanism, which allows it to weigh the importance of neighboring nodes differently rather than treating them all equally. By doing so, the model can focus more intently on the neighbors that are more informative, thereby mitigating the influence of irrelevant or noisy connections. To capture multiple facets of interactions between proteins, GAT employs several attention heads, each of which learns its own set of attention weights. The features weighted by multiple attention heads are fused, either through concatenation or averaging, to create an enriched representation of the target protein. Then, a non-linear activation function like ReLU is applied to introduce complexity into the model's understanding of the data. Finally, the enhanced feature set is used to predict the function of the protein, typically through a fully connected layer or some other suitable prediction architecture. The model aims to minimize a loss function, such as cross-entropy, by aligning its predictions as closely as possible to the true labels.	(1) GATs are capable of learning both local and global features of each node by aggregating information from its neighbors, which is crucial for understanding complex protein functions, (2) GAT uses attention mechanisms to weigh the importance of a node's neighbors. This is useful in biology where not all interactions are equally significant. The attention scores can provide insights into which proteins or protein domains are crucial for a particular function, (3) The attention mechanism also makes the model more interpretable. You can potentially understand which features are important for prediction, (4) GATs can be easily scaled to large graphs and can also be incorporated into more complex models. They also do not require the graph to be homogeneous or to have a fixed structure, (5) GATs can work with nodes that have complex features, such as amino acid sequences, 3D structures, allowing the model to make more informed decisions, (6) Biological datasets can be highly imbalanced with many negative examples. GATs can be adapted to handle such scenarios effectively, (7) GATs can learn to identify protein functions end-to-end, making the whole process more streamlined and potentially more accurate.	(1) The architecture of the GAT must be tailored to the specific needs of protein function detection. Multiple attention heads and layers can be fine-tuned to capture various relationships and hierarchies among proteins, (2) Hyperparameter tuning, including learning rates, dropout rates, and L2 regularization, can significantly affect the model's performance, (3) The training dataset should be balanced in terms of functional classes to avoid biases. It should also be sufficiently large to capture the diversity of protein functions, (4) Sufficient computational resources are essential for training a model that can cope with the complexity and size of biological datasets, (5) Convergence during training is critical. The model should be allowed to train until performance metrics plateau for best results, (6) Employing a robust validation strategy like k-fold cross-validation can ensure that the model generalizes well to unseen data, (7) Although GATs are more interpretable than many other neural network architectures, additional techniques might be needed to interpret the attention mechanisms for biological insights, (8) Incorporating prior knowledge can enhance the model's performance. This could be in the form of additional features or even constraints during training.	(1) GATs can become computationally expensive as the number of nodes or edges increases, making them less suitable for extremely large protein-protein interaction networks, (2) The attention mechanisms often require more memory, making it harder to scale to larger graphs, (3) Inaccuracies and noise in the protein interaction data can lead to incorrect predictions, (4) Lack of ground truth labels for many proteins can make training less effective, (5) While GATs aim to provide a more interpretable internal state via attention mechanisms, these can still be hard to interpret in a biological context, (6) GATs are susceptible to overfitting, especially when the amount of available data is limited, (7) The attention mechanism might not capture all the important features of the proteins, missing out on subtle interactions that could be crucial for function prediction, (8) GATs trained on one type of protein network may not generalize well to other types of protein networks, (9) The performance might be sensitive to the choice of hyperparameters, requiring a lot of tuning, (10) Proteins are dynamic entities that undergo post-translational modifications, and these temporal aspects are not captured well by static graphs.
Convolutional Neural Networks (CNN) [13-16, 62, 63]	The core operation in CNNs is the convolution, which is applied to the input data with the use of a filter or kernel. This operation captures local spatial dependencies in the data, like the arrangement of amino acids in a local region of a protein sequence or the spatial arrangement of atoms in a 3D protein structure. An activation function like ReLU is applied elementwise to introduce non-linearity into the system. Non-linearity helps the network learn from the error and adjust during the learning process. Pooling layers down-sample the spatial dimensions of the input, reducing the number of parameters and making the network computationally efficient. Max-pooling is used, which takes the maximum value in a local region of the data. At the end of the architecture, one or more fully connected layers are used to perform high-level reasoning. The output layer uses a softmax activation function for multi-class classification tasks like determining protein function. If 3D structure information is available, CNNs can identify important spatial features and patterns that traditional sequence-based methods might not capture.	(1) CNNs are adept at automatically learning features hierarchically. In the lower layers, they capture local patterns like edges and simple shapes, while higher layers capture more complex features. In the context of proteins, these layers could be responsible for capturing different levels of spatial or sequential motifs, higher-order interactions, etc., all of which are essential for determining protein function, (2) CNNs utilize shared weights and pooling layers that help the network generalize well, even to new, unseen data. This is useful in biology where data can be sparse and high-quality labeled examples are not always readily available. In proteins, certain motifs and domains that are functionally important can appear in different spatial configurations. CNNs can identify these critical features irrespective of their spatial orientation, (3) Proteins can have complex structures, and CNNs can capture this complexity by using multiple layers and various types of connections, often leading to better performance than simpler models, (4) Pre-trained CNNs can be fine-tuned for the specific task of protein function prediction, leveraging learned features from other domains or similar tasks. This can accelerate training.	(1) Rotating, flipping, and varying the scale can increase the robustness of the model, (2) The number, size, and stride of convolutional layers need to be optimized for the specific task, (3) Functions like ReLU, Leaky ReLU, and others can impact the network's learning capacity, (4) Max pooling or average pooling can reduce dimensionality and computational costs, (5) Batch size should be optimized to balance between gradient accuracy and computational resources, (6) An optimal learning rate can significantly affect the convergence speed and model performance, (7) Techniques like dropout, weight decay, and batch normalization can help in reducing overfitting, (8) Understanding which features contribute most to the decision-making process can be useful, (9) Use techniques to interpret the model's predictions, especially in life-critical applications, (10) Residue Context: Considering local and global context can improve function prediction, (11) Multi-task Learning: Models can be optimized for multiple tasks simultaneously, like predicting multiple types of protein functions, and (12) Temporal and Spatial Consistency: Ensuring the CNN model accounts for the time-evolving nature of proteins can make predictions more accurate	(1) Training CNNs on intricate, large-scale datasets is resource-intensive, needing specialized equipment and extended runtimes, (2) CNNs are frequently termed "black boxes," complicating the understanding of their decision-making, especially when the biological consequences are unclear, (3) Increasing model complexity or using limited data raises the overfitting risk, compromising the model's ability to generalize, (4) CNNs can find it tough to manage imbalances in protein function classes, frequently miscategorizing less common functions, (5) Proteins with multiple functions present a hurdle for CNNs, which are generally tailored for single-label tasks, (6) The influence of factors like tissue or cellular environment make protein function context-sensitive, a nuance traditional CNNs struggle with, (7) CNN models optimized for one data domain may not transition well to others without employing domain-specific adjustments, (8) Many CNN structures fail to consider the distinct spatial and temporal dynamics of protein function, affecting their ability to identify meaningful biological patterns, (9) For CNNs to be effective, a high-quality labeled dataset is required. The quality of protein function databases can be inconsistent.

Naïve Bayes-Based Technique	**[64-66]**	A Naïve Bayes model is trained using a dataset in which each protein's function is already identified. The model determines the likelihood of each feature occurring, given a specific class label, using Bayes' Theorem. This relationship is defined as P(Class|Features) = (P(Features|Class) × P(Class)) / P(Features). The model assumes that all features are conditionally independent when the class label is known, simplifying the calculation of P(Features|Class). When presented with a new, unlabeled protein, the model estimates the likelihood of the protein belonging to each functional category based on its features. It then assigns the protein to the functional class that has the maximum probability.	(1) Naïve Bayes is simple to comprehend and execute, (2) The technique is highly adaptable and efficient for large, ever-changing databases, like those in bioinformatics, enabling real-time analysis, (3) It accommodates a range of protein sequence features, including amino acids, dipeptide frequencies, and advanced metrics like PSSMs, (4) With its low computational demands, it's accessible to smaller labs or researchers lacking extensive computing power, (5) The algorithm tolerates some noise and irrelevant features, which are frequent in biological data, (6) Its probabilistic model enhances feature understanding and lends insight into protein biology, (7) It's effective for multi-class challenges, often encountered in protein function categorization.	(1) A Naïve Bayes classifier performs better with a substantial and balanced dataset for training. In terms of predicting protein functions, this implies having a large and varied collection of proteins whose functions are well-labeled, (2) Choosing the right features, from basic amino acid makeup to complex evolutionary data, is key for model accuracy, (3) Naïve Bayes assumes feature independence, but can still work well in biological contexts where this isn't fully met, (4) Preprocessing to remove noisy data boosts performance; in protein prediction, this means omitting poorly annotated samples, (5) A dataset that is skewed towards a particular class can result in a biased Naïve Bayes classifier. Methods like resampling and SMOTE[1] can be employed to address the issue of class imbalance	(1) With limited samples and high-dimensional data, Naïve Bayes can struggle, (2) Sparse data due to varied amino acid combinations can make Naïve Bayes' probabilities unreliable, (3) Naïve Bayes may underperform in class-imbalanced protein function categorization, especially for minority classes, (4) The algorithm ignores important contextual or sequential info, focusing on independent features, (5) Naïve Bayes can be swayed by irrelevant or noisy data, common in large biological datasets, (6) Designed for discrete data, Naïve Bayes struggles with continuous features like hydrophobicity, (7) The method falls short in capturing feature interactions in biological processes, (8) Despite its simplicity, Naïve Bayes can be computationally intense for complex data.
Decision Tree (DT)	**[6770]**	Use the training data to build the decision tree by making splits based on calculated metrics like information gain or Gini impurity. The tree will decide how to classify proteins based on these splits. To avoid overfitting, the tree might be pruned by removing branches that don't add significant power in predicting protein function. Use a separate dataset of proteins with known functions to validate the accuracy of the decision tree. Once the tree is built and validated, it can be used to predict the function of new, unknown proteins by starting at the root and traversing the tree according to the protein's features until a leaf node is reached.	(1) DTs are ideal for protein function detection due to their ease of interpretation, allowing experts to grasp the rationale behind predictions, (2) They handle diverse biological data types, like categorical amino acid types and numerical sequence lengths, efficiently, (3) DTs capture non-linear, complex feature interactions naturally, (4) They're robust to noisy or incomplete datasets, providing reasonable predictions even under imperfect conditions, (5) DTs can be easily integrated with other machine learning methods or in ensemble models for greater flexibility, (6) They are quicker to train, (7) DTs require fewer preprocessing steps, such as data normalization, making them easier to implement	(1) An imbalanced dataset could distort the decision tree's outcomes, (2) Accurate imputation of missing values is essential, (3) Deep trees risk overfitting, while shallow ones might not fully represent the data complexity, (4) Setting a low Minimum Samples Split could cause overfitting, whereas a high setting might result in underfitting, (5) Insufficient samples in a leaf node can produce unreliable classifications, (6) Metrics like the Gini index or information gain require careful selection, (7) Node-splitting algorithms can affect the decision tree's performance, (8) Tree pruning techniques can help mitigate overfitting.	(1) DT can become complex, capturing dataset noise, and leading to overfitting. This hampers their ability to work well with new data, (2) Shallow trees may not capture biological data complexity, resulting in underfitting and poor performance, (3) DT often falter with imbalanced datasets, (4) Protein data is usually high-dimensional. DT may require dimensionality reduction to perform well, (5) For large datasets, constructing a DT can demand significant computational resources, (6) DT often use binary splits, which may be inefficient for tasks like multi-class protein function prediction, (7) irrelevant features can be considered by DT, reducing accuracy, (8) DT can be sensitive to minor data changes, affecting both the model's reliability and its interpretability
Support Vector Machine (SVM)	**[71-76]**	SVM finds the optimal hyperplane that separates data points of different classes in a high-dimensional space. The hyperplane is chosen such that it maximizes the margin between the closest data points (support vectors) of different classes. SVM uses a technique known as the kernel trick to transform the input space into a higher-dimensional space, making it easier to find a separating hyperplane. This is useful when the data is not linearly separable in its original space. SVMs can be adapted to handle proteins belonging to multiple functional categories by training multiple one-vs-one. SVM allows for regularization through parameters like the cost parameter *C* and different kernel parameters to prevent overfitting.	(1) Kernel methods in SVMs transform protein features into higher dimensions, simplifying the task of finding separating hyperplanes between classes, (2) SVMs excel in generalizing, making them effective for noisy biological datasets, (3) Originally for binary classification, SVMs can adapt for multi-class problems, useful in predicting multi-functional proteins, (4) SVMs aim to find the optimal hyperplane for class separation. The support vectors can shed light on important classification features, often relevant biologically, (5) Efficient optimization techniques make SVM training computationally manageable, (6) SVMs can be easily integrated into ensemble models, providing a more robust way to predict protein functions.	(1) Select a kernel function—such as linear, polynomial, or radial basis function—that is aligned with the data's distribution and characteristics, (2) The Regularization Parameter (C) balances margin size and error rate to avoid overfitting, (3) Tune parameters like polynomial degree or gamma based on your chosen kernel, (4) Train on a dataset large enough to capture core patterns, (5) For imbalanced data, consider oversampling the minority class or tweaking misclassification costs, (6) The optimization algorithm, often SMO for SVM, affects speed and memory, (7) Use PCA for faster computations with minimal accuracy loss, (8) For scientific contexts like protein classification, use tools like LIME or SHAP for interpretability.	(1) The choice of kernel function can dramatically impact performance, (2) Kernel parameters need to be fine-tuned, often requiring exhaustive grid search, (3) If the feature vectors do not capture essential properties of proteins, classification can suffer, (4) Requires expert domain knowledge to select meaningful features, (5) Protein functions can have different frequencies, leading to biased classification towards the majority class, (6) Wrong choice of regularization parameters can lead to overfitting, (7) Noisy or incomplete data can lead the model to capture noise rather than the underlying pattern, (8) SVMs are often less interpretable, making it hard to understand feature importance, (9) SVMs trained on one type of protein data might not generalize well to other types.

[1] SMOTE (Synthetic Minority Over-sampling Technique) is a method used to generate synthetic samples from the minority class in a dataset to counteract imbalance

Extreme Gradient Boosting (XGBoost)	**[77-79]**	XGBoost works by building multiple decision trees in a sequential manner. Each tree aims to correct the errors of its predecessor. The final prediction is a weighted sum of the predictions from all individual trees. Unlike standard boosting methods, XGBoost includes L1 (Lasso) and L2 (Ridge) regularization terms in its objective function. This discourages overly complex models, reducing the chance of overfitting. While Gradient Boosting builds trees greedily, XGBoost prunes the trees as it goes, optimizing for both computational efficiency and predictive power. Missing data is quite common in biological datasets. XGBoost can automatically handle missing values, choosing the optimal splits accordingly.	(1) Proteins have multifaceted features like sequence and structure, making them high-dimensional. XGBoost excels at managing such complex data, (2) XGBoost effectively mitigates overfitting, enhancing its generalization to new data, (3) XGBoost uses an ensemble of decision trees, capturing complex relationships essential for predicting protein function, where simple linear models fall short, (4) Understanding key aspects of protein function is scientifically valuable. XGBoost offers feature importance scores for model interpretation, (5) XGBoost allows customization through various objective functions and evaluation metrics, aligning the model with specific biological queries and data types, (6) XGBoost offers tools like custom loss functions and re-sampling to adapt to diverse biological challenges	(1) Balance datasets with SMOTE for better model performance, (2) Normalize features; XGBoost scale-invariant, (3) Use PCA for feature reduction to boost performance, (4) Start with shallow trees and gradually increase the depth to find the optimum depth, (5) A lower learning rate like 0.01 or 0.1 generally works well, but it depends on the specific problem, (6) L1 (Lasso) or L2 (Ridge) regularization parameters (alpha and lambda) should be optimized, (7) Estimator count based on dataset size; use early stopping, (8) If the dataset is imbalanced, setting the class weights can help the algorithm to focus more on the minority class, (9) For high-dimensional data, setting an appropriate block size can save both memory and computation time.	(1) Protein function datasets are often imbalanced, with some functions being over-represented and others under-represented. XGBoost may not perform well on imbalanced datasets without careful tuning, (2) XGBoost can be computationally expensive, requiring significant memory for large datasets, (3) XGBoost has various hyperparameters like learning rate, max depth, etc., that need to be carefully tuned. Incorrect parameter settings can significantly affect performance, (4) Protein functions can change over time or under different conditions, which XGBoost might not capture, (5) XGBoost models generally do not consider the broader biological context in which proteins operate, such as interacting partners or cellular localization, unless such features are explicitly provided.
Random Forest (RF)	**[80-82]**	The idea is to create multiple trees during training and output the class that is the mode of the classes of the individual trees for a given input. Each decision tree in the ensemble is constructed using a random subset of the training data and a random subset of features, which helps to improve the model's generalization capability. Multiple decision trees are trained using random subsets of the data and features. Each tree gives a "vote" for classifying a new protein's function. For a new protein, each tree in the ensemble makes a prediction about its function. The final prediction is the one that gets most votes from all the trees in the forest. Due to the ensemble nature of Random Forests, they are usually more robust and accurate compared to single decision trees.	(1) RF can handle high-dimensional data without the need for feature reduction, thereby preserving the richness of the data, (2) RFs are resilient to noise in the data, (3) The relationship between protein features and their function is often non-linear. RFs can capture these non-linear relationships without assuming any specific form for the underlying model, (4) The ensemble approach reduces overfitting. This ensures that the model generalizes well to unseen data, (5) RF can compute a score indicating the importance of each feature in predicting the target variable. This is valuable where understanding which features (e.g., amino acids) are most informative can lead to biological insights, (6) RFs provides a way to understand feature importance, which is valuable in a scientific context, (7) RF is fast to train, meaning that it can handle large datasets effectively.	(1) Address class imbalance since some protein functions might be under-represented, (2) Ensure data is normalized or standardized for consistent training, (3) A higher number of trees usually increases performance but at the cost of computational power, (4) Experiment with the maximum depth of the trees to avoid overfitting, (5) Gini impurity or entropy can be chosen based on the nature of the data, (6) The minimum number of samples required to split an internal node should be fine-tuned, (7) Ensure that the model's predictions make sense in the biological context, (8) Sometimes combining Random Forest with other algorithms can improve overall performance, (9) Fine-tune the classification threshold based on the problem's sensitivity and specificity requirements, (10) Use accurate, meaningful protein features.	(1) RF can be more susceptible to overfitting on the majority class, thereby missing important minority class patterns, (2) RF provides a measure of feature importance, but this can sometimes be misleading, particularly when features are highly correlated or interdependent, (3) RF is a complex ensemble model that may be hard to interpret, (4) RF can struggle with high-dimensional data unless properly tuned, which may require significant computational resources, (5) The performance of a RF is dependent on the tuning of hyperparameters such as the number of trees, maximum depth of trees, and minimum samples per leaf. Incorrectly tuned parameters could lead to poor performance, (6) RF can be sensitive to noisy labels, (7) Training a RF on large biological datasets is computationally intensive, (8) RF can get stuck in local optima, which may not be the global optimum for the problem space
Autoencoders	**[83-87]**	The input protein data is passed through an "encoder" network, which compresses it into a "latent" or "hidden" lower-dimensional representation. This hidden layer captures the essential features needed to describe the protein. This compressed representation is "decoded" back to reconstruct the original data (to ensure that the compressed representation holds much of the original information). During training, a loss function (e.g., Mean Squared Error) quantifies how well the reconstructed output matches the original input. The neural network adjusts its parameters to minimize this loss. The encoder is used to transform new protein data into the lower-dimensional space. These compressed features can then be used to predict protein function using additional classification algorithms.	(1) Autoencoders can compress high-dimensional information into a lower-dimensional latent space, capturing essential features, (2) Autoencoders can learn meaningful representations from the existing data without requiring explicit labels. This is useful for identifying unknown protein functions, (3) The encoding process captures important features from the raw data which can be critical for functional annotation. This can be a useful step before applying supervised machine learning models for the function prediction task, (4) The reconstruction error between the original and decoded data could indicate proteins with unique or novel functions that stand out from typical protein functions, (5) The latent space learned by autoencoders can be probed to understand the important. This offers insights into what molecular features are indicative of specific functions	(1) The architecture must be apt for grasping complex relationships between protein sequences and functions, ranging from simple linear to more advanced convolutional or recurrent autoencoders, (2) To avoid overfitting, methods like dropout or regularization are useful, (3) Optimize hyperparameters like learning rate, epochs, and batch size, (4) The choice of loss function (MSE, cross-entropy, etc.) should be aligned with the goal of the task. Custom loss functions can be designed to focus on specific aspects of protein function, (5) Interpretability can be crucial in understanding how the model is making its predictions, which is particularly important in scientific contexts like this one, (6) The model can be fine-tuned or adapted to recognize specific classes of proteins or types of functions, depending on the research focus.	(1) Autoencoders may falter with sparse data, affecting accuracy, (2) They risk overfitting if the architecture is overly complex for the available data, (3) Interpreting biological significance from hidden layers is challenging, complicating understanding of predictions, (4) Scalability is a concern as biological data grows, increasing computational costs, (5) Autoencoders are less ideal for supervised tasks like protein function classification, requiring modifications, (6) Susceptible to high-level noise, affecting feature representation, (7) Training can get stuck in local optima, compromising function representations, (8) High computational requirements limit accessibility for smaller labs, (9) Performance may not generalize across different protein types or conditions, necessitating new training for each scenario.

K-Nearest Neighbor (KNN)	**[23-55, 88]**	A labeled dataset is prepared, consisting of feature vectors for proteins whose functions are already known. These serve as the "neighbors" against which new, unknown proteins will be compared. A distance metric (e.g., Euclidean distance, cosine similarity, etc.) is defined to measure the similarity between two proteins based on their feature vectors. The user must specify the number "K," which represents how many of the nearest neighbors will be considered when making a classification decision. A smaller K value like 1 or 3 might make the model sensitive to noise, whereas a larger K value could make it more resilient but potentially less accurate for individual cases. To classify the function of a new, unknown protein: (1) Compute the distance between the feature vector of the unknown protein and the feature vectors of all known proteins in the training dataset, (2) Sort these distances and select the "K" smallest ones, (3) Look at the labels (i.e., the known functions) of these K nearest neighbors, (4) Assign the most frequent label among these neighbors to the unknown protein.	(1) KNN is straightforward to implement and understand. This simplicity makes it easy to explain the model's decisions, (2) KNN makes no explicit assumptions about the underlying distribution of the data. This flexibility allows it to adapt well to the complexities and irregularities often found in biological datasets, (3) KNN is an instance-based learning algorithm that doesn't require a separate training phase. This makes it computationally less intensive to set up and allows it to adapt quickly to new data, (4) Biologically, proteins that have similar sequences or structures often perform similar functions. The KNN algorithm inherently incorporates this concept by classifying proteins based on the similarity of their features, which can include sequence motifs, structural elements, (5) Many proteins have multiple functions, and KNN can handle multi-label classification tasks natively. By considering the labels of the 'k' nearest neighbors, it is possible to predict multiple functions for a single query protein, (6) KNN can be easily combined with other machine learning algorithms or used as a component within ensemble methods to improve the overall performance of protein function prediction.	*Algorithm Configuration:* (1) The number of neighbors (*K*) should be fine-tuned. A small *K* may result in a noisy model, while a large *K* may smooth over the data too much, (2) Different distance metrics like Euclidean, Manhattan, or custom-designed metrics based on biological knowledge can affect performance, and (3) Sometimes, weighting the votes of neighbors can improve performance, especially when the distribution of classes is uneven. *Computation:* (1) KNN is computationally expensive. Data structures like KD-Tree, Ball Tree, or Approximate Nearest Neighbor methods can be used for faster query times, and (2) If possible, parallelizing the algorithm can significantly speed up computation. *Evaluation:* (1) Use k-fold cross-validation to get an unbiased estimate of the model's performance, and (2) Different metrics like precision, recall, F1-score, can be appropriate. *Domain-Specific Considerations:* (1) Sometimes features that are biologically more relevant for function prediction may not necessarily be statistically significant. A combination of domain expertise and data-driven methods is the most effective, and (2) Some proteins have more than one function, and so a multi-label classification approach may be necessary.	(1) KNN needs to store the entire dataset, making it memory-intensive, especially for large protein databases, (2) Searching for nearest neighbors can be computationally expensive for large datasets unless optimized algorithms or data structures like KD-trees are used, (3) Protein function prediction may involve high-dimensional feature spaces, which make distance measures less effective, (4) KNN is susceptible to noise in the dataset. Erroneous data points can skew predictions, (5) In the biological context, some features may be more relevant than others for protein function. KNN does not inherently weigh features, (6) The value of *K* must be chosen carefully. A small *K* can be noisy, while a large *K* can smooth out the decision boundaries excessively, (7) Choice of distance metric (e.g., Euclidean) can significantly impact performance, (8) Protein functions may be the result of complex interactions between features (e.g., amino acid sequences) that KNN might not capture effectively, (9) Some protein functions may be underrepresented, making it difficult for KNN to classify them correctly, (10) KNN is not suited for incremental learning, which is problematic if the database is constantly updated with new proteins.
Multi-Attention Mechanism	**[89-92]**	In a standard Attention Mechanism, the model calculates a weighted sum of input features by focusing on different parts. Multi-Attention extends this by using multiple 'heads' to focus on various aspects of the data at once. Each head calculates a Query, Key, and Value for every data point, like an amino acid in a protein. The Query seeks relevant information, the Key identifies focus areas, and the Value is the info to be summed. The Query from one point is matched with Keys from others to get attention scores, determining the focus each point gets. These scores help in aggregating the Values into a new data representation, better capturing protein features vital for its function. Multiple heads operate in parallel but may focus differently. Their outputs are joined and linearly projected to form the final output, which is expected to capture a richer feature set than a single head. This is often used in sequence-to-sequence models for tasks like protein classification. In advanced models, layers of multi-head attention can be stacked to learn hierarchical features crucial for understanding complex relationships in proteins.	(1) Attention mechanisms produce attention weights, which can provide insights into which amino acid residues are crucial for particular protein functions. This interpretability is highly beneficial for understanding biological mechanisms, (2) Transformers and similar architectures that use attention mechanisms are highly parallelizable during training, unlike RNNs, which must process sequences step-by-step. This efficiency can be vital when dealing with large datasets of protein sequences, (3) Multi-attention mechanisms can be adapted for various tasks related to protein function prediction, such as classification, sequence alignment, and even generative tasks for designing new proteins with desired functionalities, (4) Multi-attention allows for a richer set of interactions between features. In the case of proteins, this means being able to better understand the interplay between different types of amino acid residues and secondary structures, (5) Incorporating multi-attention mechanisms often outperform traditional models in bioinformatics tasks, including protein function prediction, (6) The attention mechanism allows the model to dynamically route information through different paths in the network during each forward pass, making it highly adaptive to the complexities found in biological sequences.	*Model Architecture:* (1) Multi-Attention often involves multiple heads to capture different aspects of the relationship between residues. The optimal number can vary, (2) The size of the hidden layers, as well as the dimensions of the attention vectors, can influence performance, and (3) The number of layers in the attention mechanism or overall network also plays a role. *Training:* (1) Techniques like dropout, layer normalization, or weight decay can prevent overfitting, (2) Batch size needs to be carefully selected for efficient training and generalization, (3) Adaptive learning rate methods like Adam may perform better, and (4) For data augmentation, techniques such as random cropping or rotation could be used, especially if the data is imbalanced. *Data:* (1) More data typically leads to better performance, although diminishing returns can be a factor, (2) Annotated data needs to be reliable; errors in the ground truth can degrade performance, and (3) Imbalanced datasets can skew the performance and make the model biased. *Computational Resources:* (1) Larger models and larger datasets will require more memory, and (2) More complex models will require more computational power, which might make experimentation slower	(1) Attention mechanisms, especially in the context of multiple layers and heads, consume a significant amount of memory, (2) Training these models on large protein databases may be computationally expensive, (3) Without proper regularization, these models can easily overfit to the training data, (4) The model might not generalize well to proteins that are significantly different from those in the training set, (5) While attention scores give some insight, it is often hard to interpret why the model makes a particular prediction, (6) The learned representations might not always align with biologically meaningful features, (7) Multi-attention models often have many hyperparameters that need to be carefully tuned, (8) The choice of the number of layers and heads can significantly impact the performance but are not trivial to optimize, (9) Even though attention mechanisms are good at capturing long-range interactions, the fixed-length context window can still limit their effectiveness for very large proteins, (10) Proteins often operate in complex, nonlinear systems, and it is not clear how well attention mechanisms can capture these dynamics, (11) The model could be sensitive to noise or errors in the data, leading to false positives or negatives.

IX. EXPERIMENTAL EVALUATIONS

In this section, we carried out experiments to examine and rank the different techniques outlined in this study. For each set of algorithms that utilizes a same technique, one representative algorithm was chosen. These representative algorithms were then methodically assessed and ranked. Every algorithm was run on a system powered by Windows 10 Pro with an Intel(R) Core(TM) i7-6820HQ processor clocked at 2.70 GHz, complemented by 16 GB of RAM.

A. Datasets for Training and Testing the Models

We obtained the CAFA3 challenge training data from September 2016 and the test benchmark from November 15, 2017, aimed at evaluating protein function prediction techniques [87, 104]. Annotations with codes like EXP, IDA, and TAS are deemed experimental. The training set includes proteins with these annotations up to September 2016, while the test benchmark covers the period until November 2017. We used the UniProtKB database from June 1, 2016, for training and testing. A blind test was conducted on 100,000 protein sequences from CAFA3. Later model assessment was enabled by newly revealed protein functions in the UniProtKB database and facilitated using the GO structure, encompassing diverse molecular, biological, and cellular categories.

Table 2. The number of protein sequences with experimental annotations in CAFA3 datasets grouped by sub-ontologies

Statistics	MFO	BPO	CCO	All
Training size	36,110	53,500	50,596	66,841
Testing size	1,137	2,392	1,265	3,328
Number of classes	677	3,992	551	5,220

B. Evaluation Metrics

We assessed the different algorithms' predictions employing the CAFA3 evaluation metrics F_{max} and S_{min} [104]. Also, we present the area under the precision-recall curve (AUPR). This metric is a suitable evaluation tool for predictions that exhibit significant class imbalances. The F_{max} represents the peak protein-centric F-measure determined across all prediction thresholds. The three metrices are defines as shown below:

- *F_{max} (Maximum F-measure):* The F-measure, a harmonic mean of precision and recall, balances the proportion of relevant instances retrieved against the total relevant. F-max, its peak value at different thresholds, optimizes this balance in protein function prediction, avoiding over-prediction and excess caution.
- *S_{min} (Minimum Semantic Distance):* S_{min} represents the smallest semantic distance across all prediction thresholds, pinpointing the threshold where predictions most closely align with true functions in biological terms.
- *Area Under the Precision-Recall curve (AUPR):* AUPR calculates the area under the curve. A model with perfect precision and recall would have an AUPR of 1.0, while a random model would typically have a much lower AUPR.

C. Hyperparameters Setting

We used the same hyperparameters as described in the original papers reported the algorithms.

D. Methodology for Selecting Representative Algorithms

Upon examining the papers that showcased algorithms using a same technique, we selected the most impactful paper. The algorithm depicted in this paper was selected to exemplify the technique (i.e., to serve as a representative of the technique). In identifying the foremost paper among all the papers that highlight algorithms employing the same technique, we weighed various criteria, such as its leading-edge nature and the date it was published. We searched for publicly available codes corresponding to the algorithms we chose as representative of their respective techniques. Of the selected algorithms, we found public codes for nine. For the other representative papers, we developed our own versions using TensorFlow, based on the methods described by Sinaga and Yang [105]. We trained these models using the Adam optimizer, in line with Sinaga and Yang's [105]. TensorFlow's APIs enable users to craft custom algorithms [106]. Our development was done in Python 3.6, leveraging TensorFlow 2.10.0 for model support. Here are the links to those codes, along with the references to their papers:

[41] https://github.com/cansyl/DEEPred
[55] https://beta.deepfri.flatironinstitute.org/
[62] https://github.com/flatironinstitute/DeepFRI
[15] https://github.com/bio-ontology-research-group/deepgoplus
[69] http://issubmission.sjtu.edu.cn/netgo/
[76] http://jing.cz3.nus.edu.sg/cgi-bin/svmprot.cgi
[82] http://www.ppved.org.cn/
[87] https://github.com/TurkuNLP/CAFA3
[88] https://github.com/VGligorijevic/deepNF

E. Ranking the techniques, sub-categories, and Categories

We employed the following strategy to rank the methods based on the outcome of the results.

- **Ranking the various techniques that belong to the same sub-category:** We computed the mean F-max, S_{min}, and AUPR scores for the selected algorithms representing the techniques that fall under the same methodology sub-category. Subsequently, we ranked the techniques within the same sub-category according to their scores.
- **Ranking the various sub-categories that belong to the same category:** We computed the mean scores for the selected algorithms representing the sub-categories that fall under the same category. Then, we ranked the sub-categories within the same category according to their scores.
- **Ranking the various categories:** We computed the mean scores for the selected algorithms representing the various categories. Subsequently, we ranked the categories according to their scores.

Figs. 3-6 show the ranking of the techniques, sub-categories, and categories based on our experimental results.

Caution: While we have chosen specific papers to represent their categories, it is important to acknowledge that this does not necessarily mean these papers are the foremost papers in their categories. The ranking we provide should be understood as a broad assessment of how each technique fares relative to others, rather than a definitive or absolute comparison.

Category	Sub-Category	Technique	Paper	Metric	Value	Tech. Rank	Sub-Categ. Rank	Categ Rank
Neural Networks	Feedforward-based	Deep Neural Networks	[41]	F_{max}	0.558	1	4	1
				S_{min}	7.275			
				AUPR	0.493			
		Multi-Layer Perceptron	[46]	F_{max}	0.533	2		
				S_{min}	7.607			
				AUPR	0.464			
	Recurrent-based	Recurrent Neural Networks	[49]	F_{max}	0.587	2	3	
				S_{min}	6.697			
				AUPR	0.525			
		Long Short-Term Memory	[51]	F_{max}	0.616	1		
				S_{min}	6.635			
				AUPR	0.561			
	Graph-based	Graph Neural Networks	[55]	F_{max}	0.643	1	1	
				S_{min}	5.583			
				AUPR	0.622			
		Graph Attention Networks	[62]	F_{max}	0.631	2		
				S_{min}	5.638			
				AUPR	0.604			
	Convolutional-based	Convolutional Neural Networks (CNN)	[15]	F_{max}	0.621	N/A	2	
				S_{min}	5.764			
				AUPR	0.583			
Statistical Inference	Probabilistic -based	Naïve Bayes-Based	[69]	F_{max}	0.356	N/A	1	4
				S_{min}	10.08			
				AUPR	0.335			
	Structural-based	Decision Tree	[72]	F_{max}	0.337	2	2	
				S_{min}	11.08			
				AUPR	0.308			
		Support Vector Machine	[76]	F_{max}	0.348	1		
				S_{min}	10.78			
				AUPR	0.327			
Ensemble Learning	N/A	XGBoost	[82]	F_{max}	0.374	2	N/A	3
				S_{min}	9.857			
				AUPR	0.364			
		Random Forest	[87]	F_{max}	0.417	1		
				S_{min}	9.079			
				AUPR	0.381			
Neighborhood -Based	N/A	Autoencoder	[88]	F_{max}	0.479	1	N/A	2
				S_{min}	7.906			
				AUPR	0.426			
		K-Nearest neighbor	[24]	F_{max}	0.424	2		
				S_{min}	8.364			
				AUPR	0.387			

(a)

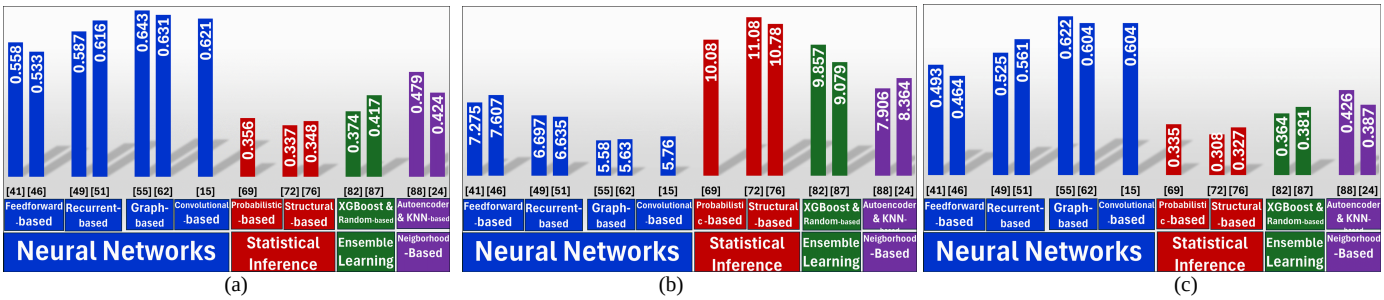
Category	Sub-Category	Technique	Paper	Metric	Value	Tech. Rank	Sub-Categ. Rank	Categ Rank
Neural Networks	Feedforward-based	Deep Neural Networks	[41]	F_{max}	0.622	1	4	1
				S_{min}	5.247			
				AUPR	0.606			
		Multi-Layer Perceptron	[46]	F_{max}	0.604	2		
				S_{min}	6.164			
				AUPR	0.573			
	Recurrent-based	Recurrent Neural Networks	[49]	F_{max}	0.671	2	2	
				S_{min}	4.808			
				AUPR	0.664			
		Long Short-Term Memory	[51]	F_{max}	0.692	1		
				S_{min}	4.632			
				AUPR	0.672			
	Graph-based	Graph Neural Networks	[55]	F_{max}	0.732	1	1	
				S_{min}	4.213			
				AUPR	0.697			
		Graph Attention Networks	[62]	F_{max}	0.708	2		
				S_{min}	4.217			
				AUPR	0.684			
	Convolutional-based	Convolutional Neural Networks (CNN)	[15]	F_{max}	0.657	N/A	3	
				S_{min}	4.955			
				AUPR	0.642			
Statistical Inference	Probabilistic -based	Naïve Bayes-Based	[69]	F_{max}	0.547	N/A	1	4
				S_{min}	9.204			
				AUPR	0.445			
	Structural-based	Decision Tree	[72]	F_{max}	0.494	2	2	
				S_{min}	10.74			
				AUPR	0.426			
		Support Vector Machine	[76]	F_{max}	0.521	1		
				S_{min}	10.07			
				AUPR	0.430			
Ensemble Learning	N/A	XGBoost	[82]	F_{max}	0.574	2	N/A	3
				S_{min}	8.253			
				AUPR	0.497			
		Random Forest	[87]	F_{max}	0.584	1		
				S_{min}	7.859			
				AUPR	0.520			
Neighborhood -Based	N/A	Autoencoder	[88]	F_{max}	0.598	1	N/A	2
				S_{min}	6.642			
				AUPR	0.554			
		K-Nearest neighbor	[24]	F_{max}	0.593	2		
				S_{min}	7.478			
				AUPR	0.545			

(b)

Category	Sub-Category	Technique	Paper	Metric	Value	Tech. Rank	Sub-Categ. Rank	Categ Rank
Neural Networks	Feedforward-based	Deep Neural Networks	[41]	F_{max}	0.522	1	4	1
				S_{min}	20.36			
				AUPR	0.503			
		Multi-Layer Perceptron	[46]	F_{max}	0.514	2		
				S_{min}	21.28			
				AUPR	0.492			
	Recurrent-based	Recurrent Neural Networks	[49]	F_{max}	0.534	1	3	
				S_{min}	19.07			
				AUPR	0.520			
		Long Short-Term Memory	[51]	F_{max}	0.526	2		
				S_{min}	19.67			
				AUPR	0.507			
	Graph-based	Graph Neural Networks	[55]	F_{max}	0.548	1	1	
				S_{min}	16.25			
				AUPR	0.536			
		Graph Attention Networks	[62]	F_{max}	0.542	2		
				S_{min}	17.87			
				AUPR	0.529			
	Convolutional-based	Convolutional Neural Networks (CNN)	[15]	F_{max}	0.539	N/A	2	
				S_{min}	18.96			
				AUPR	0.528			
Statistical Inference	Probabilistic -based	Naïve Bayes-Based	[69]	F_{max}	0.476	N/A	1	4
				S_{min}	29.40			
				AUPR	0.387			
	Structural-based	Decision Tree	[72]	F_{max}	0.461	1	2	
				S_{min}	31.84			
				AUPR	0.375			
		Support Vector Machine	[76]	F_{max}	0.456	2		
				S_{min}	33.06			
				AUPR	0.367			
Ensemble Learning	N/A	XGBoost	[82]	F_{max}	0.482	2	N/A	3
				S_{min}	26.47			
				AUPR	0.402			
		Random Forest	[87]	F_{max}	0.486	1		
				S_{min}	24.63			
				AUPR	0.409			
Neighborhood -Based	N/A	Autoencoder	[88]	F_{max}	0.514	1	N/A	2
				S_{min}	21.57			
				AUPR	0.424			
		K-Nearest neighbor	[24]	F_{max}	0.504	2		
				S_{min}	23.06			
				AUPR	0.420			

(c)

Fig 3: F_{max}, S_{min}, and AUPR scores of the selected algorithms, where: (a) using MFO, (b) using CCO, and (c) using BPO. The table also shows the following: the ranking of the techniques that belong to the same sub-category, the ranking of the various sub-categories that belong to the same category, and the ranking of the categories.



(a) (b) (c)

Fig. 4: The individual scores of: (a) F_{max}, (b) S_{min}, and (c) AUPR, using MFO dataset.

D. Discussion of the Experimental Results

DNN: It surpassed traditional methods with its convolutional layers adept at pinpointing protein motifs. This highlights the significance of amino acid sequences in protein function prediction. After 50 epochs, loss stabilized, avoiding overfitting. It had difficulty with rare protein functions.

MLP: It demonstrated high precision, showcasing its proficiency in deducing function from sequences, outperforming some other techniques. It processed data quickly, suitable for extensive proteomic studies. Its metrics showed low overfitting risk. Yet, it faltered with underrepresented protein functions and class imbalance issue.

RNN: It adepts at detecting complex amino acid sequence patterns, ensuring reliable predictions. It faced challenges with class imbalances, high computations, overfitting tendencies.

LSTM: It showed high accuracy, mainly due to its long-sequence memory. It was efficient in recognizing rare motifs and processing complex sequences. Some sequences posed challenges, especially when proteins had similar motifs. It required high computational power, lengthening training.

GNN: It excelled, achieving the highest accuracy among all. It exhibited a fine balance of precision and recall, indicating its strength in protein function prediction. The model boasted an impressive AUC-ROC score. Yet, it struggled with sparse and noisy data and demanded significant computational input.

GAT: It outdid many models in accuracy and clarity, ranking second in tested models. It demonstrated stability against minor data disturbances, making it fit for real-world tasks. It struggled predicting functions of rarely annotated proteins.

CNN: It showcased strength in understanding protein functions, enhancing their role in research and drug discovery. Its high precision indicates a deep grasp of protein sequence patterns. Its reliance on labeled data restricts full-spectrum understanding.

Naïve Bayes: Leveraging Naïve Bayes' probability, the model predicted protein functions but occasionally missed rare ones. Its assumption may not align with complex protein patterns. It's efficient, but CNNs outperform it slightly in accuracy.

DT: While fairly precise, DT's interpretability is its strong suit. Dipeptide composition is a pivotal feature. However, deeper trees risk overfitting and demand higher computational power.

SVM: SVM was effective for protein function prediction, especially with large dimensions. Its success hinges on feature vector quality. Enhancing features can improve outcomes.

XGBoost: With high precision, it outshined SVM. It trained quickly, making it fit for detailed proteomic studies. It predicted rare protein functions well. More fine-tuning can enhance it.

Random Forest: The model yielded good results in protein prediction, identifying essential sequences and structures. Its resistance to overfitting stood out. Yet, it struggled with less common protein functions.

Autoencoder: Its accuracy varied based on protein sequence complexity. It faced challenges with rarer ones. Intermediate layers provided clues for protein functions.

KNN: *KNN* offered decent results in protein function prediction. Its efficiency dropped for K values above 10. It faced hurdles in predicting diverse protein functions.

E. Discussion of the Standard in I/O Files, Metadata, and Benefit from Standardization

1) Standards in I/O Files & Metadata

RNN: Applying RNNs to the CAFA3 dataset required adapting standard sequence data formats like CSV or JSON to suit the complex nature of protein sequences, along with detailed metadata capturing the intricacies of biological sequences.

GNN: Utilizing GNN for protein function prediction involved adapting graph-structured data formats like edge lists or adjacency matrices to represent the complex relationships in biological data, with metadata describing node and edge features.

SVM: We adapted data representation away from standard CSV or LIBSVM formats to suit the unique characteristics of protein sequences, focusing on feature vectors that accurately represent biological functionalities.

LSTM: For LSTMs, we utilized formats that effectively handle the long sequences typical in the protein data from CAFA3, deviating from the CSV format used for time series or text data.

MLP: Our MLPs application to the CAFA3 challenge required a different approach to data representation, necessitating tailored formats and labeling to handle protein sequences.

Naïve Bayes: Typically used for text classification, our application of the Naïve Bayes-based technique to protein data required adaptation from standard CSV or JSON formats to formats suitable for representing complex biological features.

GAT: Like with GNNs, applying GATs to biological data required adapting standard graph formats for protein interactions' intricacies, including unique attention mechanisms

DNN: We adapted DNNs to the unique requirements of the CAFA3 and UniProtKB datasets. These protein sequence datasets demanded specialized data structuring and metadata to accurately represent protein functions.

CNN: Adapting CNNs for protein function prediction involved a departure from standard image datasets like CIFAR-10, requiring new approaches to data formatting and representation.

2) Integration of Accessory Input Data:

RNN: Our experiments showed that while integrating additional sequential data into RNNs can be challenging due to their sequential nature, advancements in RNN architectures have improved their applicability in this domain.

GNN: Our research highlighted the challenges in integrating non-graph biological data into GNNs, reflecting an active area of research in bioinformatics.

SVM: The integration of new data into SVM models depended heavily on the feature space. We found that integration within the same domain was feasible.

LSTM: Integrating additional biological data into LSTMs proved more manageable than with basic RNNs, yet there were challenges, especially with multimodal data integration.

MLP: Integrating additional biological data into MLPs was comparatively simpler due to their less complex structure, as we observed in our experiments.

Naïve Bayes: We found that integrating additional biological data, especially when it was text-based or featured similar structures, was straightforward. However, the complexity increased with multimodal data.

GAT: We faced GNN-like challenges, further complicated by attention mechanisms in a biological setting.

DNN: We found that integrating additional biological data into

DNNs varied in complexity.

CNN: We found that integrating similar biological data types was straightforward, but combining different ones was complex.

3) Benefit from Standardization

Our research highlighted the importance of standardizing data formats and preprocessing. Standardization enhanced model reproducibility, simplified experiments, and facilitated comparison of various models like MLPs and GNNs. By standardizing sequence and graph data formats, as well as feature representation and data handling practices, we could significantly improve the development, benchmarking, and comparison of the models. This not only streamlined research processes but also boosted the efficiency and accuracy of results.

X. Potential Future Perspectives for Utilizing Machine Learning in Protein Function Identification

1) Deep Neural Networks (DNN) Technique

a) DNNs can integrate with generative models like GANs to foresee modifications to proteins that may enhance or change their functionality.
b) DNNs have the capability to scrutinize the proteomes of individual patients, pinpointing protein functions linked to specific disease conditions. This enables the creation of personalized therapeutic approaches.
c) When DNNs are trained on a specific dataset, the knowledge they acquire can be applied to other similar tasks. This diminishes the requirement for vast training datasets for every distinct task.
d) DNNs can be structured to process several types of data concurrently. This includes data related to proteins' sequences, structures, and interactions, which results in precise function estimations.
e) There's a curiosity in discerning the roles of proteins from microbial communities. DNNs can annotate the functions of these metaproteomes on a grand scale.
f) DNNs can be fine-tuned to identify specific active sites on proteins. This can identify interactions like those between proteins and ligands or between proteins.

2) Multi-Layer Perception (MLP) Technique

a) Future MLP designs might consider incorporating characteristics derived from protein structures (e.g., 3D structural data) for enhanced function prediction.
b) Merging data such as PPI, gene expression patterns, and additional omics data as input features for MLPs may enhance the accuracy of function prediction.
c) Protein functionalities can be classified hierarchically (e.g., GO). By structuring MLPs in layered configurations that reflect this hierarchy, predictions could first identify broader function categories before honing in on specific functions.
d) A proactive learning method, where the model chooses which proteins should be labeled next, proves advantageous. Such repetitive fine-tuning allows the MLP to evolve and enhance over time with minimal guidance from experts.
e) By merging MLPs with other deep learning frameworks such as CNNs or RNNs, one can discern both localized and overarching trends in protein sequences.

3) Recurrent Neural Networks (RNN) Technique

a) A multi-modal approach can be employed where sequence data (ideal for RNNs) merges with structural or interactive data (best for neural network types) to get holistic predictions.
b) Conventional techniques may overlook interactions between remote amino acids in a protein sequence that are pivotal for determining its function. RNN models are adept at discerning these extensive relationships.
c) RNNs can aid in predicting the functionality of custom-crafted protein sequences (e.g., proteins with predetermined functions) , thereby refining the protein fabrication process.
d) RNNs hold the promise of deducing protein functionalities even from incomplete sequences. This is valuable for imperfectly sequenced genomes or scenarios with only fragmentary sequence information
e) The primary structure of many proteins contains vital clues about their functions. By processing these sequences with RNNs, these models might discern patterns linked to distinct functions.

4) Long Short-Term Memory (LSTM) Technique

a) Recognizing protein roles is fundamental in drug development. By using LSTM networks, one can forecast protein functionalities, assisting in pinpointing prospective protein targets for novel medications.
b) LSTMs have the potential to estimate PPI by leveraging both the sequence and the role of proteins. This prediction could pave the way to deducing possible roles for various proteins.
c) Certain areas within proteins could possess concealed or uncharted functions. Through analyzing the specific regions where an LSTM concentrates or pays attention, we can identify these obscure functional areas.
d) Merging LSTMs with the principles of transfer learning might enable the model to modify its estimations grounded in the acknowledged roles of protein domains. Hence, if a protein showcases a domain with a known function, this can shape the overall protein predictions.
e) LSTMs can be fashioned in a diverse task framework, forecasting multiple facets of a protein, ranging from its placement, affiliations, to its role in pathways.
f) Combined models, which bring together LSTMs with architectures like CNNs or transformers (e.g., BERT tailored for proteins), can yield more precise forecasts, tapping into the advantages of varied frameworks.

5) Graph Neural Networks (GNN) Technique

a) Merging diverse omics data sets, like genomics, proteomics, and transcriptomics, into graph structures offers an enriched perspective on biological phenomena. GNNs can detect intricate patterns and relationships.
b) Graph representations of phylogenetic trees and evolutionary linkages enable GNNs to highlight evolutionary trends associated with protein functionalities. This aids in inferring the potential roles of newly identified proteins.
c) GNNs, when combined with individualized protein interaction data and genetic variations, hold promise in forecasting personal vulnerabilities to diseases and determining tailored treatment responses.
d) GNNs can augment existing PPI networks by utilizing data from protein sequences, configurations, and other pertinent biological information. This can lead to predictions about unknown functions of new proteins.
e) GNNs can predict protein folding and forecast protein-compound interactions, crucial for drug innovation.

6) Graph Attention Network (GAT) Technique

a) GATs can be used to aggregate information from neighboring proteins in PPI networks to predict proteins' functions, weighing the contributions using attention scores.
b) Some proteins interact in a temporal manner, such as during a certain cell cycle phase. GATs can be extended to handle temporal graphs, capturing these dynamic interactions, and improving function prediction.
c) By understanding how mutations in a protein sequence can affect its function, we can make predictions about the functional impacts of unknown mutations. GATs can be used to predict these impacts based on the mutated protein's position and its interaction partners in a graph.
d) Combining GATs with other machine learning models, like transformers or recurrent networks, can improve prediction by catching local and long-range dependencies
e) By predicting how specific proteins function in the context of an individual's genetic makeup, GATs can aid in designing drugs tailored to individual patients.

7) Convolutional Neural Networks (CNN) Technique

a) CNNs can be fine-tuned to predict the location of active sites or binding sites in proteins, giving insights into potential protein-ligand or protein-protein interactions.
b) By considering the larger context of PPI networks, CNNs can be used to predict how changes in one protein might influence the function of interacting partners.
c) CNNs can be utilized to predict protein function directly from amino acid sequences. By sliding over a protein's sequence, CNNs can detect motifs and patterns that are indicative of specific biological functions.
d) Since protein function is closely tied to its 3D structure, combining sequence-based models with structural data can provide a more comprehensive understanding. CNNs can be trained to recognize spatial patterns in protein structures that correlate with specific functions.
e) With the integration of patient-specific genetic data, CNNs might be employed to predict the functional implications of genetic variations on protein function, leading to more personalized therapeutic approaches.
f) Combining the predictions from multiple models, including CNNs and other machine learning techniques, can enhance prediction accuracy and robustness.

8) Naïve Bayes-Based Technique

a) Future work could involve hybrid models where Naïve Bayes is combined with other methods, like neural networks, to maximize the strengths of each.
b) Many proteins have multiple functions. Techniques to adapt Naïve Bayes for multi-label classification, where a protein can belong to multiple classes, can be more deeply explored.
c) The "naïve" assumption that features are independent is a simplification. Future work can involve modifications or adaptations of the Bayes framework to consider some level of dependency between features, especially given that protein features often interact in complex ways.
d) Future Naïve Bayes classifiers can consider 3-dimensional structural features, which provide crucial insights to function
e) Future research can focus on techniques to adapt *Naïve Bayes* to handle imbalances protein function datasets.
f) By identifying which features (e.g., amino acid sequences, motifs) are most relevant for classification, researchers can refine Naïve Bayes classifiers for better performance. Techniques like mutual information or recursive feature elimination can be used to select the relevant features.

9) Decision Tree (DT) Technique

a) There's potential to combine DTs with graph-based representations of proteins. This would allow for the incorporation of PPI networks, metabolic pathways, and other relational data in predictions.
b) By integrating time-series data, DTs can potentially predict the dynamic changes in protein function over time, in response to environmental or cellular changes.
c) Combining DTs with active learning approaches can enable prioritization of proteins and the detection of their functions.
d) As more data emerges from different organisms, decision trees that can adapt to different domains (transfer learning) will be valuable. This ensures that

knowledge from well-studied organisms can aid predictions in lesser-studied ones.

e) Combining DTs with neural networks, where deep learning models can extract intricate patterns and DTs offer an interpretable layer, offers a powerful approach. This model can utilize the strengths of both paradigms

10) Support Vector Machine (SVM) Technique

a) The power of SVM comes from its kernel trick, which allows it to deal with non-linear relationships. Developing new and biology-specific kernels can make SVM even more powerful and relevant for predicting protein functions.

b) Combining SVM with features extracted from deep learning models could enhance the prediction accuracy.

c) With the rise of personalized medicine, there could be a future in which SVMs are trained on individual patient's omics data to predict protein functions specific to that individual's biology.

d) Leveraging unsupervised or semi-supervised techniques with SVM helps in extracting meaningful information from unknown or partially known protein sequences.

e) By considering proteins as nodes in a larger biological network, SVM can be combined with network-based methods for a more holistic understanding and prediction of protein function.

11) Extreme Gradient Boosting (XGBoost) Technique

a) XGBoost's inherent ability to rank features based on their importance can help researchers identify crucial amino acid sequences or structural motifs that are critical determinants of protein function.

b) There's potential to use transfer learning, saving resources. XGBoost can be a key in such strategies.

c) Combining sequence-based predictions (like amino acid sequences) with structure-based features (like 3D conformations) in an XGBoost model can significantly improve the accuracy of protein function predictions.

d) The flexibility to define custom loss functions in XGBoost allows researchers to tailor the algorithm to specific needs of protein function prediction tasks.

e) Diverse organisms are sequenced. XGBoost's ability to generalize from limited data can be instrumental in predicting the functions of rare or unseen functions.

12) Random Forest Technique

a) Protein function can vary based on cell context, location, and development stage. Adding temporal and spatial data to the RF model enhances protein prediction.

b) Deep learning methods like CNNs or RNNs can merge with RF for sequential protein data. They can extract features for the RF model.

c) Upcoming research may craft RF designs suited for biological data, similar to specific neural network models for images or texts.

d) Partnerships between research groups might yield unified, large-scale RF models. Such models trained on varied datasets can be more reliable.

e) Future research can include a wider range of protein features in the RF, giving richer context about the protein's purpose.

f) RF can leverage transfer learning, especially for organisms with scarce annotated protein data.

13) Autoencoder Technique

a) Autoencoders extract essential data features for input into another machine learning model for prediction.

b) Autoencoders, combined with architectures like transformers or RNNs, can model both spatial and temporal protein data.

c) Autoencoder models can guide exploration of proteins or functions, prioritizing potential novel insights.

d) Proteins' three-dimensional structures give function insights. Autoencoders, trained on this, identify key structural motifs indicating function.

e) Autoencoders, with custom architectures, merge sequence, structural data, and even PPI networks for comprehensive function prediction.

14) K-Nearest Neighbor (KNN) Technique

a) KNN can be integrated with neural networks. Sequences or features can be embedded in a high-dimensional space using deep learning techniques, and then KNN can be used to identify the nearest neighbors. The combination enhances function prediction accuracy.

b) KNN can be integrated with PPI networks to enhance function prediction. Proteins that interact with each other participate in the same biological processes. By mapping proteins onto such networks and considering their neighbors, KNN can provide accurate predictions.

c) Future work can focus on integrating multiple types of data, such as sequence data and structural data, in a unified manner. KNN can be extended to work in such multi-modal contexts to improve prediction accuracy.

d) With new protein data being generated continuously, KNN models can be designed to adapt and learn continuously from new data, enhancing their prediction capabilities over time.

e) KNN can be utilized in predicting functions of proteins in individual patients, aiding in customized drug design or treatment plans.

15) Multi-Attention Mechanism Technique

a) Integrating different types of data (e.g., sequence, structure, and experimental data) can be challenging. Multi-attention mechanisms can potentially focus on relevant features across different modalities,

enhancing prediction capabilities.

b) Multi-attention can enable models to capture subtle patterns and relationships across diverse protein sequences, paving the way for improved detection of functional similarities.

c) With multi-attention, models can better handle the vast variability in protein sequences, ensuring that important, functionally relevant motifs or domains are identified, even if they are present in long and diverse sequences.

d) Multi-attention can be crucial for classifying proteins into families and sub-families based on differences and similarities, thus inferring potential function.

e) Some proteins exhibit varied functions based on cellular context. Multi-attention aids in recognizing these roles by emphasizing different sequence or structure patterns.

XI. Conclusions

This article aimed to comprehensively review ML algorithms for protein function prediction, adopting a consistent technique, methodology sub-category, and overarching methodology category. With this new taxonomy, scientists can better understand and compare algorithms, identifying their merits and pitfalls. This categorization aids in steering current research, shaping the inception and critique of new algorithms. Our work advances the ML-based protein function prediction realm by offering a more organized categorization approach.

Our research delivers a holistic classification for ML algorithms and combines empirical and experimental tests to gauge method efficiency. Our empirical review evaluated ML methods for protein function prediction using four specific metrics. Experimentally, we ranked: (1) algorithms with the same technique, (2) varied methodology sub-categories within one primary category, and (3) diverse methodology categories.

Experimental findings showed that GNN and GAT models achieved the highest accuracies, while DT and SVM were least accurate. GNN was top-ranked overall, displaying strong precision and recall in protein function prediction. GAT surpassed many in accuracy, ranking second, and proved stable against data fluctuations. DT model risked overfitting and are resource-intensive. SVM's performance tied to its feature vector quality, suggesting refined features might enhance results.

References

[1] Cao, R.; Cheng, J. “Integrated protein function prediction by mining function associations, sequences, and protein–protein and gene–gene interaction networks”. Methods 2016, 93, 84–91.

[2] Liolios, K. *et al.* "The Genomes On Line Database (GOLD) in 2009: Status of genomic and metagenomic projects and their associated metadata. Nucleic Acids Res. 2009, 38, D346–D354.

[3] Ashburner, M.; Ball, C.A.; Blake, J.A.; Botstein, D.; Butler, H.; Cherry, J.M.; Davis, A.P.; Dolinski, K.; Dwight, S.S.; Eppig, J.T.; et al. “Gene Ontology: Tool for the unification of biology”. Nat. Genet. 2000, 25, 25.

[4] Rifaioglu, A. S. *et al*. “Large-scale automated function prediction of protein sequences and an experimental case study validation on PTEN transcript variants”. *Proteins Struct. Funct. Bioinf.* 86, 135–151 (2017).

[5] Doğan, T. *et al*. “UniProt-DAAC: domain architecture alignment and classification, a new method for automatic functional annotation in UniProtKB”. *Bioinformatics* 32, 2264–2271 (2016).

[6] Lan, L., *et al.* “MS-kNN: protein function prediction by integrating multiple data sources”. *BMC Bioinformatics* 14, 1–10 (2013).

[7] Wass, M. N., *et al.* “CombFunc: Predicting protein function using heterogeneous data sources”. *Nucleic*

[8] *Acids Res.* 40, 466–470 (2012).

[9] Tiwari, K. *et al.* “A survey of computational intelligence techniques in protein function prediction”. *Int. J. Proteomics* 2014, 1–22, 2014

[10] Koskinen, P., Törönen, P., Nokso-Koivisto, J. & Holm, L. “PANNZER: High-throughput functional annotation of uncharacterized proteins in an error-prone environment”. *Bioinformatics* 31, 1544–1552 (2015).

[11] A. Jain, D. Kihara, “Phylo-PFP: improved automated protein function prediction using phylogenetic distance of distantly related sequences”, Bioinformatics 35 (2019) 753–759.

[12] C. Zhang, *et al*, “COFACTOR: improved protein function prediction by combining structure, sequence and protein-protein interaction information”, Nucleic Acids Res. 45 (2017) W291–W299.

[13] Maxat Kulmanov, *et al.* “DeepGO: predicting protein functions from sequence and interactions using a deep ontology-aware classifier”, *Bioinformatics*, 34(4), 2018, Pages 660–668.

[14] V. Golkov *et al*., "3D Deep Learning for Biological Function Prediction from Physical Fields," *2020 International Conference on 3D Vision (3DV)*, Fukuoka, Japan, 2020, pp. 928-937.

[15] Kulmanov M, Hoehndorf R. “DeepGOPlus: improved protein function prediction from sequence”. Bioinformatics. 2020 Jan 15;36(2):422-429.

[16] V. Kumar, A. Deepak, A. Ranjan and A. Prakash, "Lite-SeqCNN: A Light-Weight Deep CNN Architecture for Protein Function Prediction" in IEEE/ACM Transactions on Computational Biology and Bioinformatics, vol. 20, no. 03, pp. 2242-2253, 2023.

[17] R. Cao, C. Freitas, L. Chan, M. Sun, H. Jiang, Z. Chen, ProLanGO: protein function prediction using neural machine translation based on a recurrent neural network, Molecules 22 (2017) 1732.

[18] Liu, X. L. Deep Recurrent Neural Network for Protein Function Prediction from Sequence. *arXiv* 1–38 (2017).

[19] J. Li, L. Wang, X. Zhang, B. Liu and Y. Wang, "GONET: A Deep Network to Annotate Proteins via Recurrent Convolution Networks," in 2020 IEEE International Conference on Bioinformatics and Biomedicine (BIBM), Seoul, Korea (South), 2020 pp. 29-34.

[20] Xia W, *et al.* “PFmulDL: a novel strategy enabling multi-class and multi-label protein function annotation by integrating diverse deep learning methods”. Comput Biol Med. 2022.

[21] G. Yu, K. Wang, G. Fu, M. Guo and J. Wang, "NMFGO: Gene Function Prediction via Nonnegative Matrix Factorization with Gene Ontology," in *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, vol. 17, no. 1, pp. 238-249, 2020.

[22] Y.-W. Liu, T.-W. Hsu, C.-Y. Chang, W.-H. Liao, J.-M. Chang, “GODoc: high throughput protein function prediction using novel k-nearest-neighbor and voting algorithms”, BMC Bioinf. 21 (2020) 276.

[23] Lan L, *et al.* “MS-kNN: protein function prediction by integrating multiple data sources”. BMC Bioinformatics. 2013;14 Suppl 3:S8.

[24] Törönen P, Holm L. “PANNZER-A practical tool for protein function prediction”. Protein Sci. 2022 Jan;31(1):118-128.

[25] Li Yh, *et al.* “SVM-Prot 2016: A Web-Server for Machine Learning Prediction of Protein Functional Families from Sequence Irrespective of Similarity”. PLoS ONE 11(8): e0155290, 2016.

[26] C. Zhao, T. Liu, Z. Wang, “PANDA2: protein function prediction using graph neural networks”, NAR Genom. Bioinform. 4 (2022).

[27] M. Kaleel, Y. Zheng, J. Chen, X. Feng, J.C. Simpson, G. Pollastri, C. Mooney, “SCLpred-EMS: subcellular localization prediction of endomembrane system and secretory pathway proteins by Deep N-to-1 Convolutional Neural Networks”, Bioinformatics 36 (2020) 3343–3349.

[28] J. Hong, Y. Luo, Y. Zhang, J. Ying, W. Xue, T. Xie, L. Tao, F. Zhu, “Protein functional annotation of simultaneously improved stability, accuracy and false discovery rate achieved by a sequence-based deep learning”, Briefings Bioinf. 21 (2020) 1437–1447.

[29] M. Kulmanov, F. Zhapa-Camacho, R. Hoehndorf, DeepGOWeb: fast and accurate protein function prediction on the (Semantic) Web, Nucleic Acids Res. 49 (2021).

[30] M. Kulmanov, R. Hoehndorf, “DeepGOPlus: improved protein function prediction from sequence, Bioinformatics” 36 (2020) 422–429.

[31] [5xy] Xiaojin Zhu, Zoubin Ghahramani, and John Lafferty. 2003. Semi-supervised learning using Gaussian fields and harmonic functions. In Proceedings of the Twentieth International Conference on International Conference on Machine Learning (ICML'03). AAAI Press, 912–919.

[32] 31. Anderson, J. A. *An introduction to neural networks*. (MIT Press, 1995).

[33] 32. Hinton, G. *et al*. “Deep Neural Networks for Acoustic Modeling in Speech Recognition”. *IEEE Signal Process. Mag*. 82–97, (2012).

[34] 33. Deng, L., Hinton, G. & Kingsbury, B. “New Types of Deep Neural Network Learning For Speech Recognition And Related Applications:

An Overview", 1–5 (2013).
[35] 34. Angermueller, C. *et al*. "Deep Learning for Computational Biology". *Mol. Syst. Biol.* 12, 1–16 (2016).
[36] 35. Min, S., Lee, B. & Yoon, S. "Deep learning in bioinformatics". *Brief. Bioinform.* 18, 851–869 (2016).
[37] 36. Taigman, Y., Ranzato, M. A., Aviv, T. & Park, M. Deepface 1–8, (2014)
[38] 37. Gawehn, E., Hiss, J. A. & Schneider, G. "Deep Learning in Drug Discovery". *Mol. Inform.* 35, 3–14 (2016).
[39] 38. Baskin, I. I., *et al.* "A renaissance of neural networks in drug discovery". *Expert Opin. Drug Discov. ISSN* 11, 785–795 (2016).
[40] 39. Mayr, A., *et al.* "DeepTox: Toxicity Prediction using Deep Learning". *Front. Environ. Sci.* 3, 1–15 (2016).
[41] 40. Sureyya Rifaioglu, A., Doğan, T., Jesus Martin, M. *et al.* "DEEPred: Automated Protein Function Prediction with Multi-task Feed-forward Deep Neural Networks". *Sci Rep* 9, 7344 (2019).
[42] 41. Jan Kralj, Blaz Skrlj, Ziva Ramsak, Nada Lavrac, Kristina Gruden: "DDeMON: Ontology-based function prediction by Deep Learning from Dynamic Multiplex Networks". CoRR abs/2302.03907 (2023).
[43] 42. X. Yuan, W. Li, K. Lin and J. Hu, "A Deep Neural Network Based Hierarchical Multi-Label Classifier for Protein Function Prediction," *2019 International Conference on Computer, Information and Telecommunication Systems (CITS)*, Beijing, China, 2019, pp. 1-5.
[44] 43. R. Fa *et al.* "Predicting human protein function with multitask deep neural networks", PLoS ONE (2018).
[45] 44. A. Tavanaei, *et al.*, "Towards recognition of protein function based on its structure using deep convolutional networks," *2016 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, Shenzhen, China, 2016, pp. 145-149.
[46] 45. R. Cerri, *et al.*, "Hierarchical multi-label classification for protein function prediction: A local approach based on neural networks," *2011 11th International Conference on Intelligent Systems Design and Applications*, Cordoba, Spain, 2011, pp. 337-343.
[47] 46. HMF Ashtawy. "A Comparative Study of Machine-learning-based Scoring Functions in Predicting Protein-ligand Binding Affinity", Michigan State University. Electrical Engineering.
[48] 47. B. Çarklı Yavuz, N. Yurtay and O. Ozkan, "Prediction of Protein Secondary Structure With Clonal Selection Algorithm and Multilayer Perceptron," in *IEEE Access*, vol. 6, pp. 45256-45261, 2018.
[49] 48. Cao, R.; Freitas, C.; Chan, L.; Sun, M.; Jiang, H.; Chen, Z. "ProLanGO: Protein Function Prediction Using Neural Machine Translation Based on a Recurrent Neural Network". *Molecules* 2017, *22*, 1732.
[50] 49. Noviello TMR, Ceccarelli F, Ceccarelli M, Cerulo L. "Deep learning predicts short non-coding RNA functions from only raw sequence data". PLoS Comput Biol. 2020 Nov 11;16(11):e1008415.
[51] 50. A. Ranjan, *et al*., "Deep Robust Framework for Protein Function Prediction Using Variable-Length Protein Sequences," in *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, vol. 17, no. 5, pp. 1648-1659, 1 Sept.-Oct. 2020.
[52] 51. Zhang F, *et al.* "A Deep Learning Framework for Gene Ontology Annotations With Sequence- and Network-Based Information. IEEE/ACM Trans Comput Biol Bioinform. 2021, 18(6):2208-2217.
[53] 52. J. Wekesa, *et al.*, "LPI-DL: A recurrent deep learning model for plant lncRNA-protein interaction and function prediction with feature optimization," in 2020 IEEE International Conference on Bioinformatics and Biomedicine (BIBM), Seoul, Korea (South), 2020 pp. 499-502.
[54] 53. Shen Z, Zhang Q, Han K, Huang DS. "A Deep Learning Model for RNA-Protein Binding Preference Prediction Based on Hierarchical LSTM and Attention Network". IEEE/ACM Trans Comput Biol Bioinform. 2022 Mar-Apr;19(2):753-762.
[55] 54. Gligorijević, V., *et al.* "Structure-based protein function prediction using graph convolutional networks". *Nat Commun* 12, 3168 (2021).
[56] 55. M. Li, W. Shi, F. Zhang, M. Zeng and Y. Li, "A Deep Learning Framework for Predicting Protein Functions With Co-Occurrence of GO Terms," in *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, vol. 20, no. 2, pp. 833-842, 1 March-April 2023.
[57] [1xy] M. Li, W. Shi, F. Zhang, M. Zeng and Y. Li, "A Deep Learning Framework for Predicting Protein Functions With Co-Occurrence of GO Terms," in *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, vol. 20, no. 2, pp. 833-842, 2023.
[58] [2xy] Taha K, Yoo PD, Alzaabi M. iPFPi: A System for Improving Protein Function Prediction through Cumulative Iterations. IEEE/ACM Trans Comput Biol Bioinform. 2015 Jul-Aug;12(4):825-36.
[59] 56. Ruheng Wang et al., DeepBIO: an automated and interpretable deep-learning platform for high-throughput biological sequence prediction, functional annotation and visualization analysis, *Nucleic Acids Research*, Volume 51, Issue 7, 24 April 2023, Pages 3017–3029.
[60] 57. V. Ioannidis, *et al*., "Graph Neural Networks for Predicting Protein Functions," *IEEE 8th Intern. Workshop on Comp. Advances in Multi-Sensor Adap. Processing (CAMSAP)*, Guadeloupe, 2019, pp. 221-225.
[61] 58. H Abdine, M Chatzianastasis, C Bouyioukos, M Vazirgiannis, "Prot2Text: Multimodal Protein's Function Generation with GNNs and Transformers", arXiv preprint arXiv:2307.14367, 2023.
[62] 59. Boqiao Lai , Jinbo Xu, "Accurate protein function prediction via graph attention networks with predicted structure information", *Briefings in Bioinformatics*, Volume 23, Issue 1, January 2022, bbab502.
[63] [3xy] Mostafavi, S., Ray, D., Warde-Farley, D. *et al.* GeneMANIA: a real-time multiple association network integration algorithm for predicting gene function. *Genome Biol* 9 (Suppl 1), S4, 2008.
[64] [4xy] Peña-Castillo, L., Tasan, M., Myers, C.L. *et al.* A critical assessment of *Mus musculus*gene function prediction using integrated genomic evidence. *Genome Biol* 9 (Suppl 1), S2, 2008.
[65] 60. Zihao Li, Changkun Jiang, Jianqiang Li: "DeepGATGO: A Hierarchical Pretraining-Based Graph-Attention Model for Automatic Protein Function Prediction". CoRR abs/2307.13004 (2023).
[66] 61. Baranwal, M., Magner, A., Saldinger, J. *et al.* "Struct2Graph: a graph attention network for structure based predictions of protein–protein interactions". *BMC Bioinformatics* 23, 370 (2022).
[67] 62. S. J. Giri, P. Dutta, P. Halani and S. Saha, "MultiPredGO: Deep Multi-Modal Protein Function Prediction by Amalgamating Protein Structure, Sequence, and Interaction Information," in *IEEE Journal of Biomedical and Health Informatics*, vol. 25, no. 5, pp. 1832-1838, May 2021.
[68] 63. Cai Y, *et al.* "SDN2GO: An Integrated Deep Learning Model for Protein Function Prediction". Front Bioeng Biotechnol. 2020 Apr 29;8:391.
[69] 64. You R, Yao S, Xiong Y, Huang X, Sun F, Mamitsuka H, Zhu S. "NetGO: improving large-scale protein function prediction with massive network information". Nucleic Acids Res. 2019 Jul 2;47(W1):W379-W387.
[70] 65. C. Silla and A. Freitas, "A Global-Model Naive Bayes Approach to the Hierarchical Prediction of Protein Functions," *IEEE International Conf. on Data Mining*, Miami Beach, FL, USA, 2009, pp. 992-997.
[71] 66. GW Tang, "Predicting Protein Function and Protein-Ligand Interactions Through Machine Learning", Stanford University, 2014.
[72] 67. M. Singh, P. Singh and H. Singh, "Decision Tree Classifier for Human Protein Function Prediction," *International Conference on Advanced Computing and Communications*, Mangalore, India, 2006, pp. 564-568
[73] 68. V. R. K. S. Yedida, *et al*., "Protein function prediction using decision trees," *2008 IEEE International Conference on Bioinformatics and Biomeidcine Workshops*, Philadelphia, PA, USA, 2008, pp. 193-199
[74] 69. A. J. Deen and M. Gyanchandani, "Machine Learning Classifiers based on Predicting Membrane Protein using Decision Tree and Random Forest," *2020 Fourth International Conference on Inventive Systems and Control (ICISC)*, Coimbatore, India, 2020, pp. 562-569
[75] 70. Yi Pan, "Protein Structure Prediction and Interpretation with Support Vector Machines and Decision Trees," *International Conf on Computer and Information Technology (CIT'05)*, Shanghai, China, 2005.
[76] 71. Cai CZ, Han LY, Ji ZL, Chen X, Chen YZ. "SVM-Prot: Web-based support vector machine software for functional classification of a protein from its primary sequence". Nucleic Acids Res. 2003, 1;31(13):3692-7.
[77] 72. Barot, M. "Deep learning for protein function prediction and novel class discovery", (Order No. 30315316). Available from ProQuest Dissertations & Theses Global. (2832985041), 2023.
[78] 73. A. J. Deen and M. Gyanchandani, "Machine Learning Kernel Methods for Protein Function Prediction," *Intern Conf on Smart Systems and Inventive Technology (ICSSIT)*, Tirunelveli, India, 2019, pp. 1184-1188.
[79] 74. S. Saha and P. C. Shill, "Protein Structure Prediction in Structural Genomics without Alignment Using Support Vector Machine with Fuzzy Logic," *Intern Conf on Electrical, Computer and Communication Engineering (ECCE)*, Chittagong, Bangladesh, 2023, pp. 1-6
[80] 75. Jung J, Yi G, *et al.*. "PoGO: Prediction of Gene Ontology terms for fungal proteins". BMC Bioinformatics. 2010 Apr 29;11:215.
[81] 76. S. Yadav, *et al*., "Classification of enzyme functional classes and subclasses using support vector machine," *International Conference on Futuristic Trends on Computational Analysis and Knowledge Management (ABLAZE)*, Greater Noida, India, 2015, pp. 411-417.
[82] 77. Gou X, *et al.* "PPVED: A machine learning tool for predicting the effect of single amino acid substitution on protein function in plants". Plant Biotechnol J. 2022 Jul;20(7):1417-1431.

[83] 78. Wang P, Zhang G, Yu ZG, Huang G. “A Deep Learning and XGBoost-Based Method for Predicting Protein-Protein Interaction Sites”. Front Genet. 2021 Oct 26;12:752732.
[84] 79. Daniel Kool. “Machine learning for prediction of protein properties”, Iowa State University, 2013.
[85] 80. Srivastava, Ankita, *et al*. "A comparative analysis of SVM random forest methods for protein function prediction." *Intern Conf on Current Trends in Computer, Electrical, Electron & Communication (CTCEEC)*. 2017.
[86] 81. K. Okada, *et al*, "Microenvironment-Based Protein Function Analysis by Random Forest," *2014 22nd International Conference on Pattern Recognition*, Stockholm, Sweden, 2014, pp. 3138-3143
[87] 82. Hakala K, Kaewphan S, Bjorne J, Mehryary F, Moen H, Tolvanen M, Salakoski T, Ginter F. “Neural Network and Random Forest Models in Protein Function Prediction”. IEEE/ACM Trans Comput Biol Bioinform. 2022 May-Jun;19(3):1772-1781
[88] 83. Vladimir Gligorijević and others, “deepNF: deep network fusion for protein function prediction”, *Bioinformatics*, Volume 34, Issue 22, November 2018, Pages 3873–3881.
[89] 84. L. J. Miranda and J. Hu, "A Deep Learning Approach Based on Stacked Denoising Autoencoders for Protein Function Prediction," *2018 IEEE 42nd Annual Computer Software and Applications Conference (COMPSAC)*, Tokyo, Japan, 2018, pp. 480-485
[90] 85. L. J. Miranda and J. Hu, "Feature Extraction Using a Mutually-Competitive Autoencoder for Protein Function Prediction," *2018 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, Miyazaki, Japan, 2018, pp. 1337-1342.
[91] 86. Bonetta R, Valentino G. “Machine learning techniques for protein function prediction”. Proteins. 2020 Mar;88(3):397-413
[92] 87. Dhanuka R, Tripathi A, Singh JP. “A Semi-Supervised Autoencoder-Based Approach for Protein Function Prediction”. IEEE J Biomed Health Inform. 2022 Oct; 26(10):4957-4965.
[93] [7xy] Re M, Mesiti M, Valentini G. A fast ranking algorithm for predicting gene functions in biomolecular networks. IEEE/ACM Trans Comput Biol Bioinform. 2012 Nov-Dec;9(6):1812-8.
[94] 88. Liu Yw, Hsu TW, Chang CY, Liao WH, Chang JM. “GODoc: high-throughput protein function prediction using novel k-nearest-neighbor and voting algorithms”. BMC Bioinformatics. 2020 Nov 18;21.
[95] 89. Rongtao Zheng et al., “Large-scale predicting protein functions through heterogeneous feature fusion”, *Briefings in Bioinformatics*, Volume 24, Issue 4, July 2023, bbad243.
[96] 90. Li, F., Zhu, F., Ling, X. Liu, Q. “Protein Interaction Network Reconstruction Through Ensemble Deep Learning With Attention Mechanism”. Front. Bioeng. Biotechnol. 2020, 8, 839
[97] 91. A. Ranjan, A. Tiwari and A. Deepak, "A Sub-Sequence Based Approach to Protein Function Prediction via Multi-Attention Based Multi-Aspect Network" in IEEE/ACM Transactions on Computational Biology and Bioinformatics, vol. 20, no. 01, pp. 94-105, 2023.
[98] 92. Zhongyu Wang, *et al*., “MMSMAPlus: a multi-view multi-scale multi-attention embedding model for protein function prediction”, *Briefings in Bioinformatics*, Volume 24, Issue 4, July 2023.
[99] [6xy] M. Frasca and N. C. Bianchi, "Multitask Protein Function Prediction through Task Dissimilarity," in *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, vol. 16, no. 5, pp. 1550-1560, 2019.
[100] [1k] K. Jha, S. Saha and S. Karmakar, "Prediction of Protein-Protein Interactions Using Vision Transformer and Language Model," in *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, vol. 20, no. 5, pp. 3215-3225, 1 Sept.-Oct. 2023
[101] [2k] A. Kabir and A. Shehu, "Sequence-Structure Embeddings via Protein Language Models Improve on Prediction Tasks," *2022 IEEE International Conference on Knowledge Graph (ICKG)*, Orlando, FL, USA, 2022, pp. 105-112
[102] [3k] K. Choi, Y. Lee and C. Kim, "GCL-GO: A novel sequence-based hierarchy-aware method for protein function prediction," *2022 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, Las Vegas, NV, USA, 2022, pp. 51-56
[103] [4k] D. Hoksza and H. Gamouh, "Exploration of protein sequence embeddings for protein-ligand binding site detection," *2022 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, Las Vegas, NV, USA, 2022, pp. 3356-3361
[104] 93. Radivojac,P. et al. (2013) A large-scale evaluation of computational protein function prediction. Nat. Methods, 10, 221–227.
[105] 94. Sinaga. K. and Yang, M. "Unsupervised K-Means Clustering Algorithm," in *IEEE Access*, vol. 8, pp. 80716-80727, 2020
[106] 95. Morselli, C. and Giguere, C. “Legitimate strengths in criminal networks,” *Crime, Law Social Change*, 45(3):185–200, 2006.

Kamal Taha has been an Associate Professor in the Department of Electrical Engineering and Computer Science at Khalifa University, UAE, since 2010. He received his Ph.D. in Computer Science from the University of Texas at Arlington, USA. He has over 100 refereed publications that have appeared in prestigious top ranked journals, conference proceedings, and book chapters. Over 30 of his publications have appeared in IEEE Transactions journals. He was as an Instructor of Computer Science at the University of Texas at Arlington, USA, from August 2008 to August 2010. He worked as Engineering Specialist for Seagate Technology, USA, from 1996 to 2005 *(Seagate is a leading computer disc drive manufacturer in the US)*. His research interests span bioinformatics, information retrieval, data mining, databases, information forensics & security, and defect characterization of semiconductor wafers, with an emphasis on making data retrieval and exploration in emerging applications more effective, efficient, and robust. He serves as a member of the program committee, editorial board, and review panel for many conferences and journals.