Leveraging Graph Machine Learning for Moonlighting Protein Prediction: A PPI Network and Physiochemical Feature Approach*

1st Hongliang Zhou
School of Informatics
University of Edinburgh
Edinburgh, United Kingdom
varianzhou@gmail.com

2nd Rik Sarkar
School of Informatics
University of Edinburgh
Edinburgh, United Kingdom
rsarkar@inf.ed.ac.uk

***Abstract*—Moonlighting proteins (MPs), characterized by their ability to perform multiple independent functions, play key roles in various metabolic pathways and disease mechanisms. Accurately predicting these proteins remains a challenge in the field of bioinformatics. Traditionally, state-of-the-art methods use the physiochemical features of proteins as predictors, employing machine learning approaches like Support Vector Machines (SVM), K-Nearest Neighbors (KNN), and Random Forests (RF). Recently, Graph Neural Networks (GNNs) surged with their capabilities in handling the graph-based data and found success in various bioinformatics applications. This study focuses on evaluating the efficacy of GNNs in predicting MPs. We introduce a specialized GNN-based framework, designed specifically for MP prediction. This framework employs Protein-Protein Interaction (PPI) networks, comprising query proteins and their interacting partners, as foundational graphs, incorporating physiochemical properties as node features. Through testing with two representative models—Graph Convolutional Networks (GCN) and Graph Attention Networks (GAT)—our approach not only excels in accuracy but also matches the performance of existing state-of-the-art methods in other metrics. The adaptability of the GNN architecture offers substantial potential for developing more advanced prediction techniques, likely increasing the accuracy of these models.**

***Index Terms*—Graph Machine Learning, Moonlighting Proteins, Protein-protein Interactions, Physiochemical Features, Computational Biology**

I. Introduction

Moonlighting proteins (MPs) are a special group of multifunctional proteins that perform multiple independent functions under different contexts. These proteins have been identified in a wide range of organisms, including humans, mice, and bacteria, and are key in various metabolic and disease pathways. One of their notable roles is in facilitating the invasion mechanisms of bacteria, thereby making their identification crucial for developing novel therapeutic strategies for certain diseases [1].

Despite their critical importance, the identification of MPs was challenging, primarily due to the limitations of experimental methods. The current methods for MP identification lack a systematic approach, leading to their discoveries being largely accidental. Consequently, the databases dedicated to MPs, such as MoonProt [20] and MultitaskProt [21], are notably sparse, documenting fewer than 500 proteins. This scarcity stands in stark contrast to the rapidly expanding protein sequence databases, highlighting a pressing issue: the need for effective strategies to accurately label and identify MPs.

The advent of computational methods has significantly transformed the landscape. A notable contribution in this field was made by Gomez et al., who introduced a novel approach utilizing protein-protein interaction (PPI) data, employing a combination of graph theory and bioinformatics techniques [13]. Despite providing critical insights, this method primarily focused on identifying super-multifunctional proteins, rendering its scope somewhat limited. Building on this, Khan et al. ventured into machine learning, developing a predictive model based on PPI features. However, despite the innovative approach, their success in accurately predicting MPs was relatively modest [14].

Recent advancements in the field have led to the introduction of innovative tools like MPFit and DextMP, which leverage functional annotations and text-mining technologies for the identification of moonlighting proteins (MPs). These tools have achieved an impressive accuracy rate of 98%, marking a significant leap forward in MP research [4] [5]. Despite this high level of accuracy, these methods confront substantial scalability challenges. Currently, less than 0.5% of proteins in the Protein Data Bank (PDB) are annotated, and an even smaller fraction has associated relevant literature. This limited scope of annotated and well-documented proteins renders these approaches somewhat impractical for broader application in the field.

Recognizing these limitations, recent research has shifted towards exploiting physicochemical properties, such as quasi-sequence-order (QSOrder) and amino acid composition (AAC), which are directly extracted from the protein sequences, to predict MPs [6]–[8]. One interesting phenomena was that, in this realm, it has been observed that traditional machine learning methods, like Support Vector Machine (SVM) [9], K-nearest-neighbour (KNN) [10], and Random Forest (RF) [11], surprisingly outperform Deep Neural Networks

(DNNs) [12]. This discovery has also turned away the focus in the field away from Neural Network Methods.

The introduction of Graph Neural Networks (GNN) represents a milestone in the realms of bioinformatics and machine learning [15]. GNNs stand out for their sophisticated handling of relational data, a feature that is crucial in accurately modeling the intricate networks found in biological systems. GNN frameworks, like *Graph Convolutional Networks(GCN)*, *Graph Attention Networks(GAT)* [16] [17], have been extensively applied in the prediction of protein functions, achieving considerable success. Their ability to integrate protein-protein interaction (PPI) network data with protein sequence data was particularly noteworthy. This integration has not only enhanced the accuracy of protein function predictions but has also opened new avenues for understanding complex biological interactions and mechanisms [29].

This research explores the potential of GNN models and comparing them against previous ones. We introduce a GNN framework for MP prediction. Our framework uses PPI networks as the foundational graphs, with physicochemical features as node features. We evaluated two leading GNN architectures—Graph Convolutional Networks (GCN) and Graph Attention Networks (GAT) [16] [17]—against traditional models like SVM, KNN, RF, and DNN. Using a benchmark dataset comprising 310 proteins, split into 174 MPs and 136 non-MPs, we approach MP prediction as a graph classification task [18].

The results of our study indicate that GNN models excel in various metrics, including accuracy, precision, F1 score, and ROC-AUC. A key finding was that GNNs, with their advanced capability to process relational data, effectively overcame the limitations faced by DNNs, thereby underscoring the robustness of neural network methods in this domain.

Our work substantiates the effectiveness of GNNs in predicting MPs. The flexibility of our framework lays the groundwork for the development of more domain-specific models and features, indicating significant room for improvement. Furthermore, the application of interpretability methods can shed light on the underlying mechanisms of these models, offering valuable biological insights. This approach not only enhances the predictive power but also contributes to a deeper understanding of the complex interactions and functionalities of MPs.

II. Related Works

Before discussing our method, we would like to review related fundamental concepts such that the readers can easily understand our work.

A. Graph Neural Networks

Graph Neural Networks (GNNs) provides a scheme in learning representations on graph-structured data. They extend the capabilities of traditional deep learning methods, which excel in grid-like data structures (such as images and time series), to irregular graph domains. To date, a large number of GNN frameworks have been proposed. In this section, we briefly introduce the principles and two representative models, Graph Convolutional Networks(GCN), and Graph Attention Networks(GAT), we experimented with in our study. In the next section, we introduce how to frame the MP prediction as a Graph Classification task.

1) Basic Principle of GNNs: The core idea of GNNs is to learn a representation (embedding) of each node in a graph by aggregating information from its neighbors. This process is often referred to as *message passing*, and it involves two primary steps: aggregation and update.

Consider a graph $G = (V, E)$, where V is the set of vertices (nodes) and E is the set of edges. Let $h_v^{(l)}$ be the feature representation of node v at layer l. The message-passing scheme can be formalized as:

$$a_v^{(l)} = \text{AGGREGATE}^{(l)}(h_u^{(l-1)} : u \in \mathcal{N}(v)), \tag{1}$$

$$h_v^{(l)} = \text{UPDATE}^{(l)}(h_v^{(l-1)}, a_v^{(l)}), \tag{2}$$

where $\mathcal{N}(v)$ denotes the set of neighbors of node v, and $a_v^{(l)}$ is the aggregated information from the neighbors. Different GNN frameworks vary in their aggregate and update functions.

B. Graph Convolutional Networks (GCNs)

GCNs [16] generalize the concept of convolution from regular grids to graphs. The key idea is to define a convolution operation on the graph by aggregating neighbor information. The aggregation in GCNs is typically a weighted average function. For a given node v, the update rule in a GCN layer is given by:

$$h_v^{(l)} = \sigma\left(W^{(l)} \cdot \sum_{u \in \mathcal{N}(v) \cup v} \frac{h_u^{(l-1)}}{|\mathcal{N}(v)|}\right), \tag{3}$$

where $W^{(l)}$ is a learnable weight matrix for layer l, and σ denotes a non-linear activation function, such as ReLU.

C. Graph Attention Networks (GATs)

GATs [17] introduce the mechanism of attention to GNNs. In GATs, the weights assigned to neighbor nodes are not fixed but are learned through an attention mechanism. This allows the model to focus more on important neighbors. The update rule for a single attention head in GATs is:

$$h_v^{(l)} = \sigma\left(\sum_{u \in \mathcal{N}(v)} \alpha_{vu} W^{(l)} h_u^{(l-1)}\right), \tag{4}$$

where α_{vu} is the attention coefficient, computed as a function of node v and u's features, indicating the importance of node u's features to node v. The details can be traced in *Veli ˇckovi et al.*'s work [17]

Graph Neural Networks, through their unique *message passing* paradigm, provide a versatile framework for learning on graph-structured data. GCNs and GATs represent two foundational approaches, offering both simplicity and adaptability for various applications, particularly in fields like bioinformatics. Their ability to capture complex relationships in data makes them indispensable tools in modern machine learning.

III. Methods

A. MP prediction as a Graph Classification Task

Our method frames the prediction of moonlighting proteins (MPs) as a graph classification task [18]. Formally, this task is represented as $\mathcal{T} = (\mathcal{G}, \mathbf{y})$, where $\mathcal{G}$ denotes a collection of graphs, and $\mathbf{y}$ is a binary vector of labels. Each graph g_i in $\mathcal{G}$ corresponds to a specific protein's Protein-protein Interaction (PPI) network. The associated label $\mathbf{y_i} \in \{0, 1\}$ in $\mathbf{y}$ indicates the moonlighting status of the protein, with 1 representing an MP and 0 a non-MP.

For every target protein, we construct a graph $\mathbf{G} = (\mathbf{X}, \mathbf{E})$. In this structure, $\mathbf{E}$ represents the weighted adjacency matrix that describes the PPI network. The weights in $\mathbf{E}$ reflect the strength or confidence level of each interaction. $\mathbf{X}$ is the feature matrix containing attributes of the nodes. As will be introduced in the next section, these attributes include physicochemical properties or other relevant characteristics that might affect a protein's likelihood of being an MP. Therefore, the set $\mathbf{G}$ comprises all graphs created from the target proteins using this methodology, and $\mathbf{y}$ encompasses their corresponding classification labels.

B. Graph Constructions

For graph constructions, we used the closest K interacting partners of each query protein in the PPI network to construct the graph $\mathbf{G}$. As a compromise, we limited the size of graphs to between 20 and 30 nodes. This limitation is due to the fact that larger graphs might involve more noise and fewer certain edges, whereas smaller graphs may not adequately represent interacting patterns. $\mathbf{E}$ represents the edges in the PPI networks, indicating interactions between two proteins. We assumed that all interactions are bidirectional. The edges can be either labelled or not; in our study, for simplicity, we chose to unlabel all edges and consider them identically.

C. Node Features

For node features $\mathbf{X}$, we follow previous works and use physiochemical features [8] [19]. Since QSOrder and AAC showcased competent performances on traditional ML methods [6], we primarily focused on them in our study.

1) quasi-sequence-order (QSOrder): The quasi-sequence-order (QSOrder) feature, denoted as $x(k)$, is defined based on the normalized occurrence of amino acid type k (where k ranges from 1 to 50). The formula for $x(k)$ is given by:

$$\psi(k) = \begin{cases} \frac{f_k}{\sum_{i=1}^{20} f_i + \omega \sum_{d=1}^{30} \tau_d} & \text{if } 1 \leq k \leq 20 \\ \frac{wTk-20}{\sum_{i=1}^{20} f_i + \omega \sum_{d=1}^{30} \tau_d} & \text{if } 21 \leq k \leq 50 \end{cases} \tag{5}$$

In this formula, f_k is the frequency of amino acid type k. The weighting factor is given as 0.1 in our setting. This descriptor provides a combination of sequence composition and sequence order information, with the total number of descriptor values for QSOrder being 100. τ_d is the Sequence-order coupling number, for which details can be found in Chen et al.'s work [6].

2) Amino Acid Composition (AAC): Amino Acid Composition (AAC) refers to the proportion of each amino acid in a protein sequence. Specifically, the AAC for a given amino acid i is calculated as:

$$x_i = \frac{n_i}{\sum_{j=1}^{20} n_j} \tag{6}$$

where n_i represents the number of occurrences of amino acid i, and the denominator sums the occurrences of all 20 standard amino acids in the sequence. AAC provides a normalized representation of the frequency of each amino acid in the protein.

D. Dataset

1) Moonlighting Proteins: This study employed a benchmark dataset consistent with prior research [7], [8], [19], consisting of 351 proteins, including 215 moonlighting proteins (MPs) and 136 non-moonlighting proteins (non-MPs). These proteins represent a wide array of organisms, such as humans, E. coli, yeast, rats, fruit flies, and thale cress. We sourced MP data from the MoonProt database [20], a comprehensive resource for experimentally verified MPs. To minimize redundancy and ensure data quality, we excluded proteins with more than 40% sequence similarity using CD-hit [21]. Non-MPs were selected from the MPFit dataset [4], based on stringent criteria, and this dataset was directly downloaded and processed in our study.

2) Protein-protein Interaction(PPI) Data: For Protein-protein interaction (PPI) data, the STRING database [22] was used, which includes a wide spectrum of both experimentally obtained and computationally predicted interactions. We extracted the closest interacting partners for each protein, downloading data for their 10, 20, 30, and 50 nearest proteins. These interactions, whether functional (e.g., participating in the same metabolic pathway) or physical (e.g., direct contact between proteins), were treated equivalently in our analysis. To ensure data quality, these interactions were filtered based on a confidence threshold of 0.7. Out of the 215 MPs listed in [8], 174 were present in the STRING database, covering 80.9% of the MPs. All non-MPs were also successfully matched in the database. Therefore, our final dataset included 310 proteins, consisting of 174 MPs and 136 non-MPs. We obtained PPI networks of varying sizes for each protein, from their 20 closest neighbors up to 100, to ensure a comprehensive analysis.

E. Evaluation

In our study, we used four metrics, precision, accuracy [31], F1-score, and ROC-AUC, to evaluate the performance of each

model. The equestions of these metrics are given as follows:

$$\text{ACC} = \frac{TP + TN}{TP + TN + FP + FN}$$
$$\text{Precision} = \frac{TP}{TP + FP}$$
$$\text{Recall} = \frac{TP}{TP + FN}$$
$$F\text{-score} = \frac{2}{\frac{1}{\text{Precision}} + \frac{1}{\text{Recall}}} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

Where TP (true-positive rate) represents the number of correctly predicted MPs, TN (true negative rates) represents the number of correctly predicted non-MPs, FP (false positive rate) represents the number of incorrectly predicted MPs as non-MPs, and FN (false negative rate) represents the number of incorrectly predicted non-MPs as MPs.

IV. Experiments

In our experiments, we evaluated the performance of Graph Convolutional Networks (GCN) and Graph Attention Networks (GAT) using a preprocessed dataset that had undergone normalization and feature extraction. We utilized QSOrder and AAC, along with their combination, as node features to investigate their impact on the models' performance. These Graph Machine Learning (GML) models were compared against baseline machine learning algorithms—namely, Support Vector Machines (SVM), Random Forest (RF), K-nearest Neighbors (KNN), and Deep Neural Networks (DNN). In these comparisons, the traditional models used only the physiochemical features as inputs. To support reproducibility and foster further research in this field, we have made the source code publicly accessible.

Feature	Model	ROC-AUC	Precision	Fi-score	Accuracy
QSOrder	GAT	**72.8**	**75.4**	74.5	**74.9**
	GCN	72.2	74.4	74.0	**74.4**
	KNN	70.7	71.0	**78.9**	73.1
	SVM	**73.5**	**76.1**	77.7	74.2
	RF	72.2	74.9	76.8	73.0
	DNN	69.5	72.9	74.4	70.3
AAC	GAT	72.1	74.3	73.6	73.8
	GCN	72.3	74.6	74.1	**74.5**
	KNN	71.1	71.5	**79.3**	73.6
	SVM	**73.8**	**76.5**	**77.8**	74.4
	RF	70.9	73.4	76.3	71.9
	DNN	68.6	72.1	74.0	69.8
QSorder + AAC	GAT	**73.5**	**75.5**	74.5	**74.6**
	GCN	72.4	74.9	74.2	**74.5**
	KNN	70.8	71.0	**78.9**	73.1
	SVM	**73.7**	**76.3**	**77.8**	74.4
	RF	72.4	75.3	76.9	73.1
	DNN	70.2	73.2	75.2	71.0

TABLE I

THE EXPERIMENTAL RESULTS OF THE MODELS, GAT, GCN, WITH THEIR PPI NETWORKS ENRICHED BY QSORDER, AAC, AND THEIR COMBINATION ON GCN, GAT. AND THE PREVIOUS MODELS KNN, SVM, RF, DNN BASED SOLO ON THESE FEATURES. THE TOP 5 IN EACH COLUMN IS GIVEN IN BOLD FORMAT.

A. Predicting MPs Using Only QSOrder Number

First, we downloaded the PPI network of all proteins with their 30 closest neighbors, then we utilized the QSOrder vector of each protein within the PPI networks as the exclusive feature for the feature matrix. This approach was inspired by the methodology of Shirafkan et al. [8], and we used the R-package ftrCOOL [23] to facilitate feature extraction.

While deriving the QSOrder feature from proteins in the networks, there are 11 proteins bearing unidentified amino acids, labeled as 'X'. Such occurrences are often attributed to instrumentation or experimental limitations. A closer analysis revealed this data loss is insignificant, so we proceed the experiment with these ambiguous segments excluded.

In the initial phase of our study, we applied the GCN and GAT models to our dataset, followed by traditional methods—SVM, KNN, and RF— with only QSOrder features for predicting MPs in 310 proteins. We subsequently evaluated a DNN model configured with two layers, each comprising 256 neurons.

To address challenges posed by the limited dataset size, we adopted a comprehensive strategy to enhance model robustness and mitigate performance variability:

1) Data Preparation: The dataset was randomized to prevent model bias, setting aside a 15% subset as an independent test set. The remaining data underwent 10-fold cross-validation, providing a robust assessment of model reliability and safeguarding against overfitting.
2) Class Balance: Stratified sampling ensured equal representation of MPs and non-MPs, crucial for maintaining class balance. We further stratified graphs by connectivity—very sparse (¡ 0.25), sparse (0.25–0.5), dense (0.5–0.75), and very dense (0.75–1.0)—to preserve consistency in class and connectivity distributions across folds.
3) Model Evaluation: Model Evaluation: To counteract variability across folds, we instituted a thresholded ensemble approach. Models achieving a validation accuracy above 0.7 were included in an ensemble, applying a soft voting mechanism on the test set. This selective ensemble method integrated only the most reliable models, enhancing predictive reliability.

For the GCN and GAT models, EarlyStopping was implemented to monitor validation loss, halting training after 25 epochs without improvement. The learning rate was dynamically adjusted, and halved after 10 epochs lacking performance gains, starting with an initial rate of 0.001. We capped the training iterations at 500, utilizing the Adam optimizer [24]. Binary cross-entropy served as the loss function, fitting for our binary classification task. Edge information was encoded in a binary edge matrix, denoting interaction presence. Our models, crafted using PyTorch-Geometric [25], adopted a three-layer structure with 256 nodes each, incorporating a 0.3 dropout rate to deter overfitting. To prevent oversmoothing [26], we limited our architecture to three layers, employing a max pooling readout function for graph-level output, which then

passed through a linear layer with an argmax function for classification. An illustration of the framework is provided in 1

Baseline models were developed using scikit-learn's default settings [27], providing a standard for comparison. The DNN was similarly constructed with scikit-learn and included EarlyStopping as a regularization measure. Our findings are promising. Table 1 illustrates that the GAT model stands out as the top performer, achieving the highest accuracy at 74.9%. It also ranks second in both precision and ROC-AUC, with scores of 75.4% and 72.8%, respectively. The GAT model's F1 score is competitive at 74.5%, narrowly behind traditional machine learning methods such as RF (76.8%), SVM (77.7%), and KNN (78.9%). Similarly, the GCN model shows robust performance with an accuracy of 74.4%, slightly edging out the SVM's 74.2%. Its F1 score (74.0%), precision (74.4%), and ROC-AUC (72.2%) are also notable, showcasing its ability to rival, and potentially surpass, conventional methods. In line with previous research, the DNN model exhibits the least impressive performance across almost all metrics, which is markedly lower than its counterparts, likely due to the constraints of the limited dataset size.

B. Incorporation of AAC

Upon integrating Amino Acid Composition (AAC) as a feature—a feature shown to be competitive in Hu et al.'s work [7]—all models were reevaluated. The secone part of Table IV shows that while the GAT model retained a competitive edge with an accuracy of 73.8%, F1 score of 73.6%, precision of 74.3%, and ROC-AUC of 72.1%, these figures reflect a decrease compared to its performance with QSOrder features.

Conversely, the GCN model saw performance gains from AAC integration, achieving the highest accuracy at 74.5% and the second-highest precision at 74.6%. Its F1 score and ROC-AUC also remained strong at 74.1% and 72.3%, respectively.

The SVM model emerged as a standout with AAC, achieving the highest F1 score of 77.8%, precision of 76.5%, and ROC-AUC of 73.8%, while coming in a close second in accuracy at 74.4%. These results suggest a particular synergy between AAC features and the SVM algorithm. Similarly, the KNN model showed slight improvements, further indicating the potential alignment of AAC with these traditional algorithms. In contrast, the RF and DNN models experienced a downturn in performance, dropping by 0.5-2% across all metrics, suggesting a less effective integration of the AAC feature with these methods.

C. Combination of AAC and QSOrder

To improve the predictive accuracy of our models, we investigated the effects of combining quasi-sequence-order (QSOrder) and Amino Acid Composition (AAC) features through concatenation. Table 1 reports the outcomes, revealing that for GAT and GCN models, this integrative strategy generally led to modest performance enhancements across multiple metrics. Contrary to expectations, however, the GAT model's accuracy decreased slightly by 0.3% with the addition of AAC, and similarly, the SVM's ROC-AUC and Precision metrics fell by 0.2%. While these declines are statistically insignificant, they introduce an important caveat: the efficacy of combining physicochemical properties is not guaranteed and requires careful consideration. It suggests that the interplay between different feature types can be complex, and their collective influence on model performance is not always straightforward.

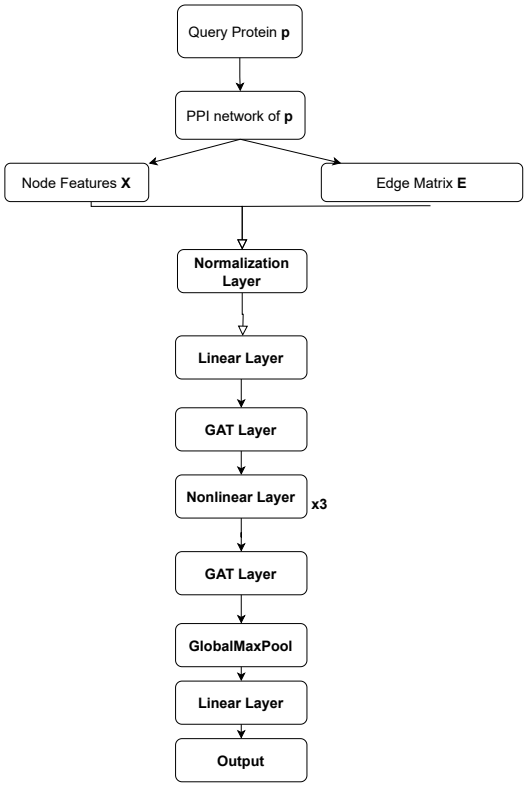

Fig. 1. An illustration of our proposed framework, Here we use GAT layers for example, but it can be replaced by any GNN layer available. For instance, for the experiment of GCN models, we replacte the *GAT layer* block with *GCN layer*.

The other models, detailed in the third section of Table 1, displayed small but uniform improvements upon utilizing the combined feature set, lending support to the idea that a careful, strategic combination of features is integral to optimizing computational tools for MP prediction. These insights point to the potential of a nuanced approach to feature selection, highlighting the need for a strategic feature synergy in enhancing the performance of models.

D. Graph Size and Model Performance

We also investigated the relationship between the size of each query protein's interaction graph and the accuracy of our predictions. Specifically, we varied the sizes of the networks sourced from the STRING database, testing limits of 20, 30,

50, and 100 nodes. We found that graphs with around 30 nodes yielded optimal performance; increasing the size beyond this point did not lead to significant improvements and, in some instances, even resulted in lower performance. Therefore, for consistency and efficiency in our experiments, we standardized on a graph size of 30 nodes, with a minimum threshold of 24 nodes. With these parameters, 85% of the 310 proteins are finally used in the training and testing of GML models.

V. Discussion

A. Potential of GNN

Our study validates the efficacy of Graph Machine Learning (GML) in predicting moonlighting proteins (MPs), with Graph Convolutional Networks (GCN) and Graph Attention Networks (GAT) demonstrating significant promise. These GML models, even in their rudimentary forms, pave the way for the development of advanced, domain-specific models that could yield deeper insights into MP behavior.

B. Comparison with DNNs

An intriguing phenomenon is that in both our study and the work conducted by Shirafkan et al. [8], Deep Neural Networks (DNNs) displayed the least effective performance in predicting moonlighting proteins (MPs), achieving a maximum accuracy of only 71%. We hypothesize that this limitation is primarily due to the data-starved nature of the DNN models. Specifically, these models are under-parameterized when dealing with the limited amounts of labeled MP data available, leading to their suboptimal performance.

Contrastingly, within a Graph Machine Learning (GML) framework, the neural network models are trained differently. In this context, each node in the network is trained once per epoch, and each node comes with its distinct set of features. Consequently, the neural network model in the GNN framework is exposed to a substantially larger number of training samples compared to traditional DNNs. This increased exposure to diverse training data contributes to the enhanced performance of GNNs in making real predictions. In further works, explanation tools and train tracking methods can be employed to compare the behavioral and connectivity differences between DNN and GNN models.

C. The implication of PPI networks

Our study supports the hypothesis proposed by Khan and Kihara [14] that Protein-Protein Interaction (PPI) networks are indicative of moonlighting proteins (MPs). This is particularly evident from the performance of Graph Attention Networks (GAT), which effectively prioritize influential nodes and pathways, thereby underscoring the significance of interaction patterns in identifying MPs. Although the mechanisms underlying GAT's effectiveness warrant further investigation, our findings suggest that focusing on the interacting partners of MPs and advancing protein embedding techniques could substantially enhance the accuracy of MP predictions.

The potential of GNN models in bioinformatics extends beyond their predictive capabilities. The field has witnessed the emergence of various explainers and tools aimed at elucidating the decision-making processes of neural networks [32]. In our future work, we intend to implement explanation tools like GNNExplainer, PGExplainer, etc. [33], [34] to demystify the operational principles of the GNN model. This endeavor is anticipated to not only provide a clearer understanding of how GNNs function but also to yield insightful biological discoveries. By dissecting the model's reasoning processes, our goal is to attain a more profound comprehension of the complex relationships and patterns within PPI networks, which are essential for accurate MP identification. Such an understanding could lead to more focused and efficacious strategies in protein function prediction and other bioinformatics applications.

D. Limitations

However, there are several limitations in our study. The most prominent one is the limited data size. Due to the limited data resources on STRING, our dataset only includes 174 MPs, covers only 81% of the 215 MPs in the previous benchmark dataset. In further work, these missed data can be imputed by alternative methods like computational PPI predictions or collected from other PPI databases. Moreover, while focusing on QSOrder and AAC features has proven beneficial, we recognize that incorporating additional features, such as structural characteristics and evolutionary information, could offer more comprehensive insights. This calls for future work to expand the feature set. Lastly, in this study, we didn't look deep into the specific types of interactions, which are potentially informative. For further work, we can delve into the nature of these interactions and involve these details to our model. With these in mind, the promising results of GML models on MP prediction are just the beginning. Further theoretical and practical refinement of these models, including exploration into other graph classification methods, are needed to enhance our understanding and prediction of MPs.

VI. Conclusion

Our investigation has showcased the potential of Graph Machine Learning (GML) for the prediction of moonlighting proteins (MPs). By harnessing physicochemical properties such as quasi-sequence-order (QSOrder) and Amino Acid Composition (AAC) in conjunction with local Protein-protein Interaction (PPI) network data, GML models, including Graph Attention Networks (GAT) and Graph Convolutional Networks (GCN), have delivered robust performance. These models have not only excelled in accuracy but also have shown superior precision, ROC-AUC, and F1-score metrics, outstripping a wide array of traditional approaches, including Random Forest (RF) and K-nearest Neighbors (KNN). Notably, the GAT model has even surpassed the accuracy of the Support Vector Machine (SVM), which is recognized as the state-of-the-art method. This is particularly significant considering that GML models have also addressed the challenges posed by data scarcity that often limit the efficacy of Deep Neural Networks (DNNs).

Our findings suggest a promising direction for the use of neural networks in MP prediction, offering a new paradigm that could reshape future research in this domain. However, our journey is not complete. The principal challenge is the small size of the existing dataset—a limitation well-known to hinder the efficacy of data-dependent neural networks. To mitigate this, our future endeavours will involve leveraging newly proposed computational PPI prediction methods alongside diverse PPI datasets to enrich the training and enhance the robustness of our GML models [28]. In addition, this study only used the simple implementation of two GML models and treated the prediction of MPs as a graph classification task. In our next stage, we plan to explore other frameworks, problem settings, and feature extractors, and involve specific interaction types, to develop a more domain-specific and robust model. Furthermore, we intend to probe the theoretical foundations of the GAT model to improve its interpretability. Our goal is to gain a more granular understanding of the mechanisms underlying MP predictions. Such efforts will not only consolidate the practical applicability of our model but also deepen our understanding of the MP and its connection with PPI.

REFERENCES

[1] Jeffery C J 1999 Moonlighting proteins *Trends Biochem. Sci.* **24(1)** 8-11

[2] Copley, S. D. (2003). Enzymes with extra talents: moonlighting functions and catalytic promiscuity *Curr. Opin. Chem. Biol.* **7(2)** 265-272

[3] Huberts D H, and van der Klei I J 2010 Moonlighting proteins: an intriguing mode of multitasking *Biochim. Biophys. Acta Mol. Cell Res.* **1803(4)** 520-525

[4] Khan I, McGraw J, and Kihara D 2017 MPFit: computational tool for predicting moonlighting proteins *Methods Protoc.* **1611** 45-57

[5] Khan I K, Bhuiyan M, and Kihara D 2017 DextMP: deep dive into text for predicting moonlighting proteins *Bioinformatics* **33(14)** 83-91

[6] Chen C, Chen L X, Zou X Y, and Cai P X 2008 Predicting protein structural class based on multi-features fusion *J. Theor. Biol.* **253(2)** 388-392

[7] Hu J and Du Y 2023 Predicting moonlighting proteins from protein sequence Proc. 16th Int. Joint Conf. Biomedical Engineering Systems and Technologies (BIOSTEC 2023) vol 3 pp 270–275

[8] Shirafkan F, Gharaghani S, Rahimian K, Sajedi R H, and Zahiri J 2021 Moonlighting protein prediction using physico-chemical and evolutional properties via machine learning methods *BMC bioinformatics* **22(1)** 1-14

[9] Hearst M A, Dumais S T, Osuna E, Platt J, and Scholkopf B 1998 Support vector machines *IEEE Intell. Syst. Appl.* **13(4)** 18-28

[10] Peterson L E 2009 K-nearest neighbor *Scholarpedia* **4(2)** 1883

[11] Breiman L 2001 Random forests *Mach. Learn.* **45** 5-32

[12] LeCun Y, Bengio Y, and Hinton G 2015 Deep learning *nature* **521(7553)** 436-444

[13] Gomez A, Hernandez S, Amela I, Piñol J, Cedano J, and Querol E 2011 Do protein–protein interaction databases identify moonlighting proteins? *Mol. Biosyst.* **7(8)** 2379-2382

[14] Khan I K, and Kihara D 2016 Genome-scale prediction of moonlighting proteins using diverse protein association information *Bioinformatics* **32(15)** 2281-2288

[15] Xu K, Hu W, Leskovec J, and Jegelka S 2018 How powerful are graph neural networks? Preprint arXiv/1810.00826

[16] Kipf T N and Welling M 2016 Semi-supervised classification with graph convolutional networks Preprint arXiv/1609.02907

[17] Veličković P et al. 2017 Graph attention networks Preprint arXiv/1710.10903

[18] Tsuda K, and Saigo H 2010 Graph classification *Manag. Min. Graph Data* **40** 337-363

[19] Chen Y, Li S, and Guo J 2022 A method for identifying moonlighting proteins based on linear discriminant analysis and bagging-svm *Front. Genet.* **13** 963349

[20] Chen C et al. 2021 MoonProt 3.0: an update of the moonlighting proteins database *Nucleic acids Res.* **49(D1)** 368-372

[21] Hernandez, Sergio, et al. "MultitaskProtDB: a database of multitasking proteins." Nucleic acids research 42.D1 (2014): D517-D520.

[22] Fu L, Niu B, Zhu Z, Wu S, and Li W 2012 CD-HIT: accelerated for clustering the next-generation sequencing data *Bioinformatics* **28(23)** 3150-3152

[23] Szklarczyk D et al. 2021 The STRING database in 2021: customizable protein–protein networks, and functional characterization of user-uploaded gene/measurement sets *Nucleic acids Res.* **49(D1)** 605-612

[24] Amerifar S, Norouzi M, and Ghandi M 2022 A tool for feature extraction from biological sequences *Brief. Bioinform.* **23(3)** 108

[25] Kingma D P, and Ba J 2014 Adam: A method for stochastic optimization Preprint arXiv:1412.6980

[26] Fey M, and Lenssen J E 2019 Fast graph representation learning with PyTorch Geometric Preprint arXiv/1903.02428

[27] Chen D, Lin Y, Li W, Li P, Zhou J, and Sun X 2020 Measuring and relieving the over-smoothing problem for graph neural networks from the topological view Proc. AAAI Conf. on artificial intelligence vol 34 No. 04 pp 3438-3445

[28] Pedregosa F et al. 2011 Scikit-learn: Machine learning in Python *J. Mach. Learn. Res.* **12** 2825-2830

[29] Soleymani F, Paquet E, Viktor H, Michalowski W, and Spinello D 2022 Protein–protein interaction prediction with deep learning: A comprehensive review *Comput. Struct. Biotechnol. J.* **20** 5316-5341

[30] Réau, M., Renaud, N., Xue, L. C., Bonvin, A. M. (2023). DeepRankGNN: a graph neural network framework to learn patterns in protein–protein interfaces. Bioinformatics, 39(1), btac759.

[31] Goutte, C., Gaussier, E. (2005, March). A probabilistic interpretation of precision, recall and F-score, with implication for evaluation. In European conference on information retrieval (pp. 345-359). Berlin, Heidelberg: Springer Berlin Heidelberg.

[32] Kakkad, J., Jannu, J., Sharma, K., Aggarwal, C., Medya, S. (2023). A Survey on Explainability of Graph Neural Networks. arXiv preprint arXiv:2306.01958.

[33] Ying, Z., Bourgeois, D., You, J., Zitnik, M., Leskovec, J. (2019). Gnnexplainer: Generating explanations for graph neural networks. Advances in neural information processing systems, 32.

[34] Luo, D., Cheng, W., Xu, D., Yu, W., Zong, B., Chen, H., Zhang, X. (2020). Parameterized explainer for graph neural network. Advances in neural information processing systems, 33, 19620-19631.