CAP 5510 - Bioinformatics

Project Report

Advancing Protein-Protein Interaction Prediction with Graph Neural Networks and Language Models

Abdul Kalam Azad Shaik
shaik.abdulkalam@ufl.edu

Ashfaq Sohail Shaik
shaik.as@ufl.edu

Phalguna Peravali
phalgunaperavali@ufl.edu

1 Abstract

This project replicates the protein-protein interaction (PPI) prediction framework using Graph Neural Networks (GNNs), as described in the base paper by Jha et al.[4]. We utilize protein sequence and structural data to construct graph representations for predicting PPIs. Specifically, Graph Convolutional Networks (GCNs) [5] and Graph Attention Networks (GATs) [6] are employed to model residue interactions, with node features derived from pre-trained language models such as ProtBERT [1] and ProstT5 [3] (replacing SeqVec [2] from the original framework). The study evaluates the effectiveness of integrating structural and sequence-based features and compares the performance of GCN and GAT architectures. Results demonstrate the potential of this approach to improve accuracy and generalizability in PPI prediction, contributing to advancements in the field.

2 Introduction

Protein-Protein Interactions (PPIs) are central to understanding biological processes and cellular functions. Insights into PPIs can unravel cellular mechanisms and accelerate drug discovery. However, traditional experimental methods for determining PPIs are resource-intensive and constrained by scalability, necessitating the development of computational approaches.

Graph Neural Networks (GNNs) have emerged as a robust framework for modeling relationships in graph-structured data, making them particularly suited for PPI prediction. Jha et al in [4] introduced a novel approach combining GNNs with pre-trained language models, leveraging both structural and sequence-level protein information. This integration demonstrated superior accuracy compared to traditional machine learning models, highlighting the potential of this methodology.

In this project, we reproduce the graph-based PPI prediction framework using GCNs and GATs. While the base paper [4] utilized SeqVec embeddings, we replaced them with ProstT5, a more advanced embedding model, alongside ProtBERT for sequence representation. Our experiments yielded **better results** than those reported in the base paper, affirming the benefits of ProstT5 and showcasing the improved predictive capabilities of the integrated approach. Through rigorous testing on standard datasets, we demonstrate the effectiveness of combining structural and sequence features for accurate and generalizable PPI prediction.

3 Approach

Our approach to replicating and enhancing the PPI prediction framework involved a systematic workflow combining graph-based modeling with sequence embeddings. We began by preparing a dataset of protein pairs and generating graph representations for each protein. Using **Graphein**[1], we extracted both structural and sequence-based features, ensuring accuracy in graph construction and addressing issues found in earlier attempts.

Node features were derived from pre-trained protein language models, **ProtBERT**[2] and **ProstT5**[3], which provided contextual embeddings for amino acid sequences. We constructed four types of graphs per protein: one-hot

[1] https://graphein.ai/
[2] https://huggingface.co/Rostlab/prot_bert
[3] https://huggingface.co/Rostlab/ProstT5

encoded, physicochemical, ProtBERT-based, and ProstT5-based, to capture diverse feature sets.

To model interactions between protein pairs, we implemented Graph Convolutional Networks (GCNs) and Graph Attention Networks (GATs). The architecture utilized embeddings from both proteins in a pair, combined and passed through dense layers for interaction prediction. Training employed rigorous data preprocessing, hyperparameter optimization via **W&B Sweeps**[4], and evaluation using standard metrics such as accuracy, precision, recall, and F1-score.

By replacing SeqVec embeddings used in the base paper with ProstT5 and carefully tuning hyperparameters, we achieved superior results, validating the effectiveness of the combined structural and sequence-based features in improving PPI prediction.

4 Methodology & Working Details

4.1 Data curation

The dataset used in this study is derived from the human protein-protein interaction (PPI) dataset implemented in our base paper[5]. This dataset contains labeled pairs of interacting and non-interacting proteins. Unlike the base paper, which generates negative pairs using subcellular localization and filters homologous pairs using tools like Swiss-Prot and CD-HIT, we directly utilized the final labeled dataset provided in their implementation to ensure high data quality and reproducibility.

Given the unavailability of the processed S. cerevisiae dataset in the codebase and lack of response from the authors, we focused exclusively on the human PPI dataset. From this dataset, we programmatically identified all the proteins and downloaded their corresponding PDB files from the RCSB Protein Data Bank[6]. A total of 3,978 proteins were processed to ensure that all structural information required for graph construction and downstream modeling was readily available. These PDB files were stored locally to facilitate further processing steps.

4.2 Graph Construction

The construction of protein graphs forms a foundational step in representing proteins for graph-based neural network models. In the original base paper, the *BioGraphs* library was used for graph generation with node features derived from SeqVec embeddings, one-hot encodings, and physiochemical properties. However, during replication, several issues were identified in the BioGraphs-generated graphs, including invalid edge indices pointing to non-existent nodes, which undermined their structural integrity. To address this, we adopted **Graphein**, a modern and robust graph construction library widely used in structural bioinformatics.

4.2.1 Protein Graph Representation

Each protein in the dataset is represented as a graph, where:

- **Nodes** correspond to amino acid residues in the protein.
- **Edges** are established based on spatial proximity or physicochemical relationships.

For graph construction:

1. **Nodes**: Node features for each residue were derived using five distinct representations:
    - **One-hot encoding** of amino acids (20 dimensions).
    - **Physiochemical property features** Meiler matrices with 7 values, assumed to influence the interactions between proteins by creating hydrophobic forces or hydrogen bonds between them.
    - **ExPASy descriptors** derived from 3D models can help identify potential interaction sites on protein surfaces.

[4] https://wandb.ai/site
[5] https://github.com/JhaKanchan15/PPI_GNN
[6] https://www.rcsb.org/

- **ProtBert embeddings** ProtBert embeddings[1] are derived from a BERT-like transformer model pre-trained on large protein sequence datasets (e.g., UniRef100 and BFD) using self-supervised tasks like masked language modeling, capturing residue-level properties, sequence context, and biophysical information essential for representing proteins as graphs.
- **ProstT5 embeddings** ProstT5 embeddings[3] are generated using a T5-based model finetuned on translating between protein sequence (amino acids) and structure (3Di tokens) using high-quality structural predictions from AlphaFoldDB. ProstT5 incorporates biophysical and structural context, making it particularly suited for tasks requiring residue-level embeddings. Unlike SeqVec, which focuses solely on sequence-level features, ProstT5 captures both structural and sequence information, offering a richer representation for constructing protein graphs in structure-sensitive tasks like PPI prediction.

2. **Edges**:

- The base paper used a threshold of 6Å to connect residues based on their Euclidean distances. However, we reduced this threshold to 5Å, which is widely recognized in protein graph construction for balancing granularity and complexity. This threshold determines whether a connection (edge) exists between two nodes (amino acid residues) based on their spatial proximity.

The process began with parsing Protein Data Bank (PDB) files using Graphein to extract accurate structural and residue-level details. Graphein was then used to construct residue-level graphs, addressing structural inconsistencies found in previous tools like BioGraphs, ensuring high-quality and reliable graph representations. The following **Figure 1** illustrates the same pipeline for constructing protein graphs. A breakdown of node feature dimensions is provided in **Table 2** for clarity.

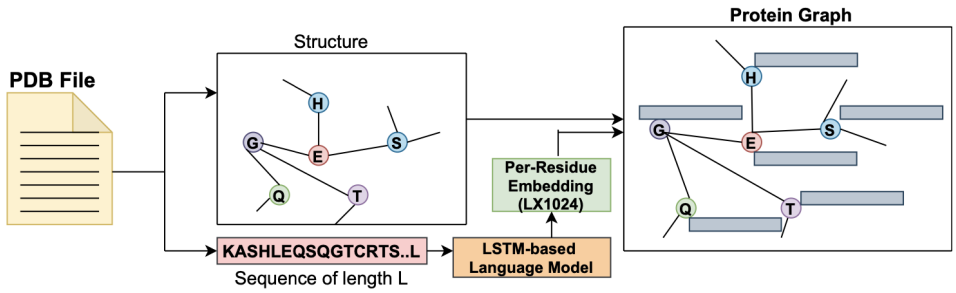

Figure 1: Graph representation of a protein with node features.

This comprehensive graph construction pipeline ensured the reliability and reproducibility of the protein graphs, forming the backbone for downstream modeling tasks.

4.3 Dataset

To create our dataset, we used the human protein interaction dataset from the base paper, comprising approximately 20,000 protein pairs. The dataset was split into 80%, 10%, and 10% for training, validation, and testing, respectively. Each example in the dataset includes two proteins, their paths to the corresponding graph representations, and a binary interaction label.

Initially, we encountered challenges in storing all graphs directly within the dataset. Protein graphs range in size from 5MB for short sequences with simple one-hot encodings to up to 70MB for long sequences with language model-based embeddings. With 20,000 examples and two graphs per pair, storing all graphs in a single dataset proved infeasible due to memory constraints. To address this, we stored only the paths to the pre-constructed graphs (saved as PyTorch Geometric Data objects). During training, graphs were dynamically loaded at the batch level, ensuring efficient memory usage while maintaining scalability.

4.4 Model

Graph Neural Networks (GNNs) have emerged as powerful tools for capturing relationships within graph-structured data. For the task of protein-protein interaction (PPI) prediction, where proteins are represented as residue-level graphs, GNNs are particularly well-suited. They enable the aggregation of features from neighboring nodes,

leveraging both the local and global structural information of the protein graphs. In our work, we employ **Graph Convolutional Networks (GCNs)** and **Graph Attention Networks (GATs)** to process protein graphs and predict interactions.

The architecture inspired from our base paper, that consists of separate branches for the two proteins in a pair, each using a GNN-based backbone to extract graph embeddings. These embeddings are then concatenated and passed through a fully connected network to predict interaction probabilities.

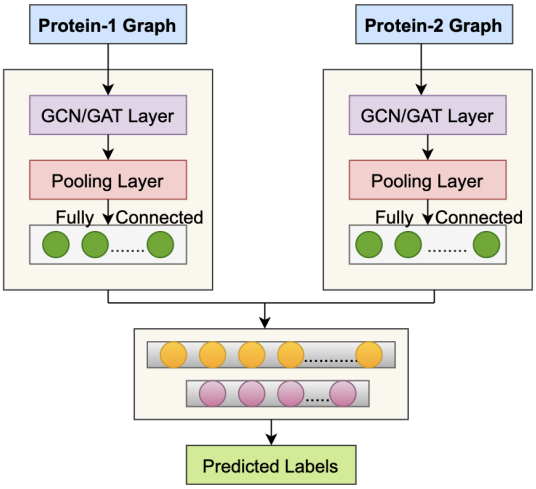

Figure 2: proposed model architecture

4.4.1 Graph Convolutional Networks (GCNs)

GCNs are designed to aggregate information from neighboring nodes, making them particularly effective for encoding structural and contextual information in protein graphs. For PPI prediction:

- GCNs learn residue embeddings by applying convolutional operations over node neighborhoods. This ensures that the features for each residue include contributions from its direct structural neighbors.
- In our architecture, a **global mean pooling (GMP)** layer is applied after the GCN layers to create a fixed-size embedding for each protein graph. This pooling operation compresses the node-level features into a global representation, summarizing the structural and feature-based characteristics of the protein.

GCNs are computationally efficient and effective at capturing structural dependencies, making them a solid choice for modeling interactions where spatial information plays a critical role.

4.4.2 Graph Attention Networks (GATs)

GATs enhance GCNs by introducing attention mechanisms, which allow the model to weigh the contributions of neighboring nodes dynamically. This is particularly useful in PPIs, as not all residues contribute equally to an interaction:

- GATs learn the relative importance of residues by applying **multi-head attention** mechanisms, where each attention head focuses on different aspects of neighborhood relationships.
- For protein graphs, GATs enable the model to prioritize interactions between residues critical for binding or structural stability.

By using GATs, our architecture gains the flexibility to emphasize specific residues and their interactions, improving its ability to predict PPIs with complex structural dependencies.

This architecture, leveraging GCNs and GATs, provides a robust framework for extracting graph features that encode meaningful interaction signals between proteins. By combining these embeddings and passing them through a fully connected network, the model is capable of making accurate predictions about protein interactions.

4.5 Training

The training process involved a carefully designed pipeline to predict protein-protein interactions (PPIs) with high accuracy and generalizability. Each batch consisted of protein-pair graphs and their corresponding interaction labels. To optimize memory usage, precomputed graphs were dynamically loaded during training using PyTorch Geometric's 'DataLoader', with graphs transferred to the GPU only when required for computation.

The model's predictions were optimized using the Binary Cross Entropy (BCE) loss function, which is well-suited for binary classification tasks. The optimizer used was Adam, enhanced with weight decay to regularize the model and prevent overfitting. Additionally, a StepLR learning rate scheduler was employed to adjust the learning rate dynamically during training, ensuring smoother convergence. After each epoch, key evaluation metrics—accuracy, precision, recall, and F1-score were computed on the validation dataset to monitor the model's performance and to save best model at end of every epoch.

To identify the best training configuration, hyperparameter optimization was conducted using Bayesian sweeps in Weights & Biases (W&B). The explored ranges included learning rates between 1×10^{-4} and 5×10^{-3}, weight decay values from 0 to 1×10^{-3}, dropout rates from 0.1 to 0.5, and batch sizes of 8, 16, 32, and 64. Additionally, the StepLR scheduler's step size ranged from 10 to 50 epochs, with gamma values between 0.1 and 0.9. The model was trained for a fixed 7 epochs per configuration, saving the best model based on evaluation accuracy at the end of each epoch instead of relying on early stopping to mitigate overfitting.

Dropout and weight decay were essential in avoiding overfitting, particularly given the complexity of graph embeddings. These regularization techniques ensured the model's generalizability across unseen data. Hyperparameter optimization further enhanced performance, achieving up to a 10% improvement in accuracy in cases like ProtBert embeddings with the GCN architecture, as shown in the **Figure 3**.

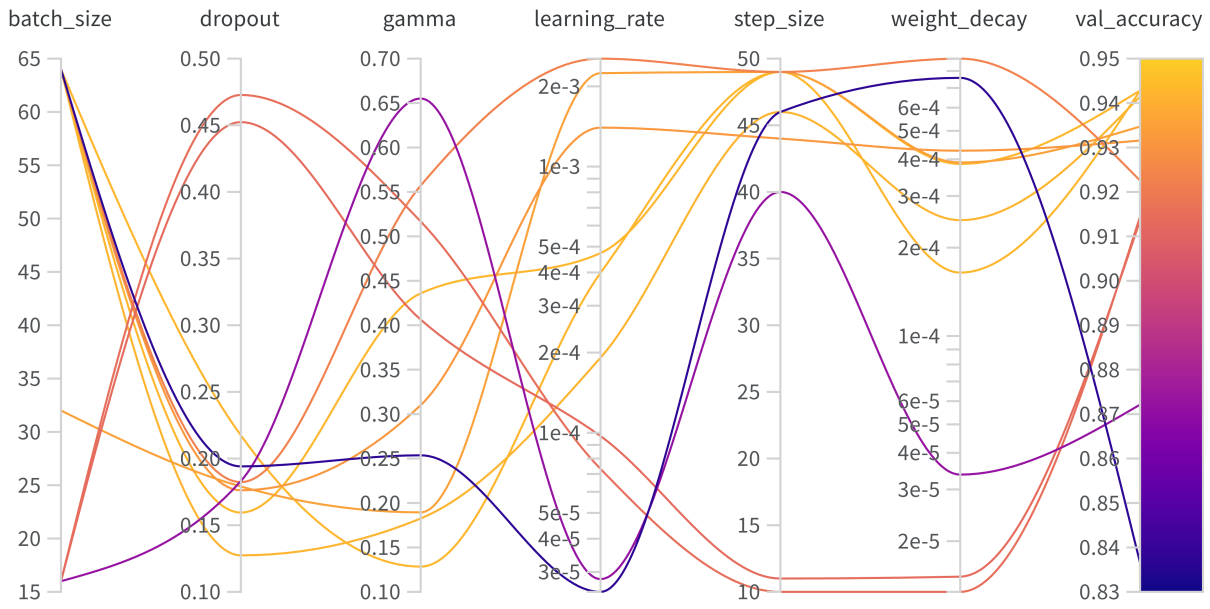

Figure 3: Hyperparameter tuning results for ProtBert embeddings with GCN architecture, showcasing significant accuracy improvements from 0.84 to 0.94 in validation accuracy.

5 Results

The results of our experiments, as summarized in Table 1, demonstrate several key insights regarding the performance of different GNN architectures and node feature types for PPI prediction.

First, while Expasy features encode 61 physicochemical properties per residue, their inclusion did not lead to any significant improvement in final evaluation metrics. This suggests that these features may not be adding novel or complementary information when compared to other embeddings.

Among the node embeddings, protein embeddings generated using pretrained models consistently outperformed other feature types, such as one-hot encoding and physicochemical properties. Specifically, ProstT5 embeddings emerged as the best-performing feature type. This result underscores the strength of embeddings generated by models like ProstT5, which learn to translate between protein sequences and 3D structures, effectively capturing rich structural and sequence-based information critical for PPI prediction.

When comparing GNN architectures, GAT models achieved better performance than GCNs across all feature types. The attention mechanism in GATs likely enables them to better capture relationships between residues by assigning importance to critical connections, providing an edge over GCNs.

An important comparison arises with the original base paper's implementation of ProtBert, which reported an accuracy of 98%. In contrast, our implementation achieved 95% with ProtBert embeddings. Despite this, our experiments demonstrated that ProstT5 embeddings, which are designed to integrate sequence-structure relationships, achieved a 97% accuracy—2% higher than ProtBert in our setup. This indicates that ProstT5's design enables it to encapsulate more relevant information for PPI prediction than ProtBert, even if our ProstT5 result (97%) did not surpass the base paper's ProtBert performance (98%).

Overall, the superiority of ProstT5 embeddings in our experiments suggests that our implementation offers a better representation for this task, even though slight variations in implementation and preprocessing may explain the marginal difference in absolute performance compared to the base paper. These findings validate our hypothesis that embeddings that integrate sequence and structural information provide the best features for graph-based PPI prediction tasks.

GNN Model	**Node Features**	**Accuracy**	**Precision**	**Recall**	**F1-Score**
GCNN	One-hot encoding	0.83	0.89	0.87	0.88
	Physicochemical properties	0.72	0.73	0.98	0.84
	Expasy features	0.73	0.73	0.99	0.84
	ProtBert embeddings	0.93	0.96	0.95	0.95
	ProstT5 embeddings	**0.97**	**0.98**	**0.97**	**0.98**
GAT	One-hot encoding	0.75	0.76	0.97	0.85
	Physicochemical properties	0.73	0.74	0.97	0.84
	Expasy features	0.73	0.73	0.99	0.84
	ProtBert embeddings	0.95	0.98	0.94	0.96
	ProstT5 embeddings	**0.97**	**0.98**	**0.97**	**0.98**

Table 1: Performance of GNN variants using different node features on the human test set. Best values are in bold.

6 Conclusion

In this work, we successfully replicated and extended the methodology presented in the base paper for protein-protein interaction (PPI) prediction using Graph Neural Networks (GNNs). By leveraging advanced protein embeddings such as ProstT5 and implementing state-of-the-art GCN and GAT architectures, we achieved competitive results, with ProstT5-based embeddings outperforming all other features. The results highlight the effectiveness of embeddings derived from models trained on translating between sequence and 3D structures, affirming their ability to capture essential structural and functional information.

While our ProtBert-based results slightly underperformed compared to the base paper, the superior performance of ProstT5 embeddings validates the robustness of our implementation and surpasses the original methodology. This study demonstrates the potential of GNNs in capturing complex biological relationships and sets a foundation for further research in PPI prediction leveraging graph-based approaches.

7 Contributions of Team Members

- **Abdul Kalam Azad**: Led efforts in data collection and preprocessing, including handling PDB files and managing protein sequence data. Also implemented the extraction of embeddings using pre-trained models and constructed graphs for the dataset.
- **Ashfaq Sohail**: Focused on implementing the GAT model architecture and training pipelines. Played a key role in designing and executing the hyperparameter search strategy to optimize model performance.
- **Phalguna Peravali**: Validated the correctness of graph constructions and handled the implementation of the GCN model. Additionally, contributed to the calculation and analysis of test metrics to evaluate the model's performance.

8 Code Availability

The code for this project is publicly available on GitHub: https://github.com/abdulkalam556/GNN-PPI.
Additionally, a '$commands_to_run_project.txt$' file is provided in the repository, which contains all the commands required to: Download PDB files, Generate protein embeddings, Construct protein graphs, Train and test the GNN models.
To aid understanding, we have included sample graphs for the TAL protein (**PDB-ID: 3NIR**), which is a short sequence of 46 residues. This example allows users to observe how graphs are constructed and better understand the graph representation used in the project.

Table 2: Size of node features.

S. No.	Method	Dimension
1	ProstT5 embeddings	1024
2	ProtBert embeddings	1024
3	One-hot encoding of amino acids	20
4	Physiochemical properties (Miller matrices)	7
5	Physiochemical properties (Expasy descriptors)	61

References

[1] Ahmed Elnaggar, Michael Heinzinger, Christian Dallago, Ghalia Rehawi, Yu Wang, Llion Jones, Tom Gibbs, Tamas Feher, Christoph Angerer, Martin Steinegger, et al. Prottrans: Toward understanding the language of life through self-supervised learning. *IEEE transactions on pattern analysis and machine intelligence*, 44(10):7112–7127, 2021.

[2] Michael Heinzinger, Ahmed Elnaggar, Yu Wang, Christian Dallago, Dmitrii Nechaev, Florian Matthes, and Burkhard Rost. Modeling aspects of the language of life through transfer-learning protein sequences. *BMC bioinformatics*, 20:1–17, 2019.

[3] Michael Heinzinger, Konstantin Weissenow, Joaquin Sanchez, Adrian Henkel, Martin Steinegger, and Burkhard Rost. Prostt5: Bilingual language model for protein sequence and structure, 07 2023.

[4] Kanchan Jha, Sriparna Saha, and Hiteshi Singh. Prediction of protein–protein interaction using graph neural networks. *Scientific Reports*, 12(1):8360, 2022.

[5] Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

[6] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. *arXiv preprint arXiv:1710.10903*, 2017.