UNIVERSIDADE DE LISBOA
FACULDADE DE CIÊNCIAS
DEPARTAMENTO DE INFORMÁTICA

Combining Deep Learning and Knowledge Graphs in PPI network prediction tasks

Laura de Oliveira Lopes Henchman Balbi

MESTRADO EM BIOINFORMÁTICA E BIOLOGIA COMPUTACIONAL

Dissertação orientada por:
Professora Doutora Catia Luísa Santana Calisto Pesquita

2022

Acknowledgments

I'd like to begin by thanking my supervisor, Professor Catia Pesquita, for the guidance and the constant availability and readiness to discuss novel experiments, which is but a symptom of excitement and genuine interest in the work. I am also very obliged to Rita Sousa for her promptness in discussing ideas, always kindly sharing some of her knowledge with me from a time when I was still figuring out the very basics of my work until now. Secondly, but not least, I thank Pedro for showing genuine interest in my work, sharing some insight on his areas of interest, some of which I now share thanks to him, and always feeding onto newer work ideas.

I take this moment to express my genuine admiration for all of my colleagues at LiSeDa and overall the members of Lasige. They brew an environment filled with ambition and curiosity, which are two must-have qualities for anyone who wishes to thrive in any field of science.

I dedicate this thesis work to my family and friends for supporting and encouraging me during this ride. Particularly to my mother for the immense care and support she gave me, not only during these past two years of studies but for the entirety of my upbringing, being the best role-model one could have as a mother, a human being and as an excellent professional.

To my brothers, Alex and Max, I apologize for not having been as present as I would have liked to. Time does not wait for work to end. I will work on making up to them from now on, so I won't miss out on so much. My apologies to my closest friends for the many times I left them on stand-by while trying to reach work goals. I recon this has only made us stronger.

Abstract

Many bioinformatics problems pertain to large, highly complex amounts of biological data that are often modelled in a graph-like arrangement to allow for a systemic-level analysis of data. Graphs provide a means of encoding biological knowledge into a formal structure that is useful for modelling and analysing relationships in biological systems. Several machine learning approaches have been developed to deal with data represented as graphs, namely by using graph neural networks, end-to-end representation-learning methods that can directly learn from graph-structured data.

A form of knowledge representation that allows for the conceptualization and specification of domains of interest is the use of ontologies. These can have a biological application into representing and structuring existing knowledge by its meaning and relationships, allowing for the organisation of large volumes of biological entities into knowledge graphs. With both biological data and biological knowledge represented as graphs, an opportunity to directly enrich the data graph with knowledge about its entities arises.

Thus, the aim of this work was to explore different approaches into combining a PPI network with information pertaining to the Gene Ontology to leverage the additional biological knowledge in a protein function prediction setting. Two methodologies were proposed for the development of said approaches. The first comprehends the creation of approaches to merging a PPI network with protein function information and combination of said approaches with different ML models to test if they benefit from the additional information during protein function prediction. The second methodology sees the construction of a GNN -based method to learning PPI and GO representations in separate and combining them for a global learning of protein function -related information. The evaluation of the different approaches and different experimental conditions with a benchmark dataset showed an overall performance increase.

Keywords: Knowledge Graphs   Ontologies   Graph Neural Networks   Protein-Protein interaction Networks

Extended Summary

Diversos dos problemas atuais no contexto da bioinformática relacionam-se com o tratamento do tamanho e complexidade dos dados biológicos, que, para serem trabalhados e analisados necessitam ser estruturados de uma forma que facilite a sua análise, uma representação gráfica. Os grafos facultam uma estrutura que capta e representa o conhecimento envolvido nos dados, sendo muito úteis para a descrição e análise de relações em sistemas biológicos, pelo que conseguem modelar diversos processos biológicos.

É ainda possível descrever e conferir significado contextual aos dados biológicos recolhidos através da integração do conhecimento disponível para o domínio biológico. A interdependência entre entidades e entre estas e os conceitos que as descrevem leva à criação de grafos de conhecimento. Assim, os grafos de conhecimento, ou "knowledge graphs"(KG), são grandes redes de entidades do mundo real, que as descrevem de acordo com os seus tipos semânticos, propriedades e inter-relacionamentos, e podem abranger vários domínios. Grandes volumes de entidades biológicas têm sido representados e organizados em grafos de conhecimento, servindo-se de uma ou várias ontologias biológicas para representar e estruturar o conhecimento biológico através do significado dos seus conceitos e relações. Neste trabalho considera-se que uma ontologia consiste na descrição semântica exata, com linguagem específica, de vários conceitos e das relações existentes entre os mesmos dentro de um determinado domínio de conhecimento, conjuntamente com os axiomas que representam condições pertinentes ao domínio da ontologia. Quando estas condições são respeitadas, permitem inferir factos que não se encontram explícitos na ontologia.

A prospeção de dados, "data mining" em inglês, é o processo de aplicação de métodos inteligentes para explorar grandes e complexos conjuntos de dados na busca de padrões que sejam úteis para detectar relações pertinentes entre variáveis, permitindo apreender o conhecimento nelas implícito. A utilização de métodos de prospeção de dados em grafos de conhecimento permite a captação de conhecimento relativo às instâncias neles descritas, que pode ser útil para tarefas de previsão sobre as mesmas.

Foram já propostas diversas formas para aplicar algoritmos de aprendizagem automática ("machine learning", ML) a tarefas de prospeção de dados sobre grafos de conhecimento, quer através do uso de representações vetoriais de baixa dimensão para aprender a informação semântica dos diversos elementos do grafo, quer através da construção de um certo tipo de modelos de aprendizagem de representações ("representation-learning "em inglês) que consigam receber diretamente e analisar dados sob a forma de grafos.

Com efeito, as redes neuronais emergem de outros métodos de ML exatamente por serem modelos de aprendizagem de representações que podem receber dados complexos e, de entre as diversas features dadas como input, conseguirem fazer uma seleção interna das mais importantes, com base na sua capacidade de representar o conhecimento que pode ser depreendido dos dados. Este conhecimento é depois apreendido através da computação de diversos parâmetros internos e da correção dos pesos associados às conexões da rede neuronal. A rede neuronal sobre grafos é um tipo particular de rede cuja arquitetura foi desenhada para ser diretamente aplicável a dados estruturados sob a forma de grafo.

De entre os conjuntos de dados do domínio público, é possível encontrar alguns contextualmente relevantes para o domínio biológico, contendo, por exemplo, informação relativa a associações entre proteínas obtidas experimentalmente ou por inferência. Contudo, a maioria não utiliza o conhecimento atual existente sobre as entidades que contêm. Quando tanto dados biológicos como o conhecimento disponível do seu domínio são representados sob a forma de grafo, surge uma oportunidade de complementar o conjunto dos dados com conhecimento sobre as suas entidades. A intenção deste trabalho é testar se e como é que complementar dados biológicos com o conhecimento atual sobre os mesmos poderá permitir que diferentes tarefas de prospeção de dados em grafos aproveitem esta informação adicional para melhorar a sua performance de previsão, em particular em redes de interações proteína-proteína (PPI).

Assim, este trabalho descreve o desenvolvimento de duas metodologias distintas: i) uma abordagem que visa a inclusão de uma camada ontológica num grafo de dados PPI, dando origem a um grafo holístico, de forma a explorar a informação funcional sobre proteínas que está representada na ontologia; ii) um método que conjuga uma arquitetura neuronal que recebe dois grafos de input com um modo de aprendizagem global que explora separadamente a informação PPI e a informação da GO para cada proteína.

Para a primeira metodologia foram exploradas diversas formas de ligar diretamente o grafo da Gene Ontology (GO) a um grafo representativo de uma rede PPI através da inclusão de diferentes conjuntos de anotações funcionais das proteínas. Foram também explorados diferentes tipos de atributos (”features”) de proteínas, conjugando features baseadas em PPI com uma medida de informação relativa às classes da GO a que as proteínas estão anotadas (ICseco). Estas experiências com diferentes tipos de anotações e de atributos geraram vários grafos holísticos, com os quais foi combinada uma variedade de métodos de ML sobre grafos pertencentes ao estado da arte. Entre estes algoritmos de ML estão incluídos tanto: i) modelos que exploram a estrutura dos grafos a partir da sua conjugação com um método de aprendizagem de representações (node2vec) para as proteínas do grafo holístico; como ii) modelos de ponta a ponta (”end-to-end”), que são capazes de aprender as suas próprias representações das proteínas (redes neuronais sobre grafos - GNNs).

Na segunda abordagem foi desenvolvido um modelo neuronal capaz de aprender em separado duas representações relativas à informação de PPI de uma dada proteina e dos aspetos funcionais que lhe são conhecidos através de duas redes convolucionais distintas. Estas representações foram posteriormente combinadas num modelo geral capaz de fazer a previsão de funções da proteína que ainda estavam por descobrir.

A avaliação das duas metodologias propostas foi realizada para uma tarefa de classifica-ção múltipla, onde o intuito era prever uma ou mais funções para as proteínas da rede. Foi utilizado um conjunto de dados de referência (OGBn-proteins), de uma rede PPI disponível para uso público. As bases de referência da primeira abordagem foram estabelecidas através da replicação de resultados publicados online de testes elaborados sobre o mesmo conjunto de dados de referência com os mesmos modelos de redes neuronais utilizados neste trabalho; a base de referência da segunda abordagem foi estabelecida através da avaliação de uma rede convolucional sobre o grafo PPI utilizado nas experiências. Para além disso, aferiu-se a capacidade dos modelos: i) fazerem previsões para proteínas de espécies deles desconhecidas, através de uma avaliação em que as proteínas são divididas por espécie pelos conjuntos de treino, validação e teste; e ii) preverem funções para proteínas cuja espécie foi dada aos modelos durante o treino, numa avaliação onde as proteínas são distribuídas aleatoriamente pelos conjuntos de treino, validação e teste. A qualidade das previsões dos modelos foi avaliada através da métrica de ROC-AUC, com a *Accuracy* como métrica auxiliar.

Os resultados obtidos demonstraram que as várias experiências desenvolvidas para cada uma das metodologias foram capazes de ultrapassar os resultados das respetivas bases de referência. Foi igualmente verificado que os modelos foram capazes de aprender informação relevante tanto para a previsão funcional de proteínas de espécies conhecidas como para prever para proteínas de espécies desconhecidas pelo modelo.

Assim, através deste trabalho, foram desenvolvidas diversas maneiras de explorar a informação funcional disponível sobre as proteínas de uma rede de interações proteina-proteina de modo a melhorar o poder de alguns dos modelos neuronais mais conhecidos da atualidade de prever aspetos funcionais desconhecidos de proteínas. Foi também construído um método baseado em GNNs capaz de combinar representações de dois aspetos distintos da caracterização das proteínas (informação PPI e informação funcional incluída na GO) para aprender um modelo geral que utiliza conhecimento contextual para prever as funções das proteínas.

Palavras-chave: Grafos de Conhecimento Ontologias Redes Neuronais em Grafos Redes de Interação Proteina-Proteina

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The impact of data availability has allowed for meaningful scientific advance and technological development. Biological research has mainly become very data-oriented; and this has raised an intricate connection to the ability to handle such huge amounts of data, while seeking to unravel novel knowledge from them. Recent high throughput-based scientific experimentation methods are able to detect thousands of protein-protein interactions (PPIs) with a single experiment (Ito et al. (2001);Joos and Bachmann (2005)). These result in the generation of enormous datasets that could very well describe the overall proteomic functional organisation. Since proteins often perform different roles in alternate biological contexts, and considering the complexity of biological systems, many of these alternate functions may not have yet been discovered. As a result, a major challenge in modern biology is to develop efficient methods for wide scale determining of protein functions and machine learning (ML) techniques have been increasingly successfully employed in this regard.

Protein-protein interactions (PPI) are one of the most commonly employed data sources for protein function prediction. Proteins oftentimes interact with other proteins present in their surroundings to execute their functions and carry out biological activities (Ryan and Matthews (2005)), so functions of unknown proteins may be discovered by leveraging the knowledge of their associations with other proteins that have already known functions. Utilising these interactions would be, therefore, invaluable to drawing new representations for proteins and understanding their roles in the mechanisms and functions of how cells work (Nariai et al. (2007)).

PPI are often represented as a graph, where proteins are the nodes and their interactions are the edges. When compared to traditional ML approaches, the application of ML on graphs allows for the learning of not only features from each of the graph's entities but also of the relationships between them, which provides valuable descriptive information of the data (Han (2002)). However, the complexity of graph data has imposed significant challenges on existing algorithms. Several approaches have been made to apply machine learning models to data represented on graphs for the inductive acquisition and generalisation of useful knowledge to make accurate predictions on unknown data. Be it either through the use of dense low-dimension vector representations (embeddings) of the different graph elements, or by building the models as end-to-end representation-learning approaches that can directly receive and analyse such graph-structured data, e.g., graph neural networks. The latter are usually built based on neural networks, and adopt a similar structure to graphs, with nodes serving as artificial neurons, and edges as weighted connections.

Recent years have seen several machine learning approaches attempting to solve biologically relevant prediction problems such as protein function prediction through PPI network analysis, such as in Gui et al. (2015), as it has been shown that interacting proteins tend to be functionally related (Lv et al. (2015)).

However, these works tend to model protein function prediction as a simple multi-class classification problem ignoring that proteins perform a multitude of functions and that the functions themselves are related to each other. Biomedical ontologies, such as the Gene Ontology (Ashburner et al. (2000)), are able to represent the complex panorama of protein function, by defining functions and how they are related to each other in a computationally amenable way. When ontologies structure large volumes of data, they form a Knowledge Graph (KG). A knowledge graph is a multi-relational graph, with different types of edges carrying different meaning that uses an ontology to describe its nodes, the entities it represents, and defines precise semantics for the relations between said entities. Combining PPI networks with a protein annotation KG affords novel opportunities for protein function prediction that can explore existing knowledge about protein function to improve performance.

1.1 Problem Definition and Objectives

Most current approaches proposed to solve prediction problems on PPI networks account for networks being presented as graph-structured data where proteins and interactions are represented as graph components. However, considering all the relevant knowledge available on each of these components, the integration of said knowledge with the data, e.g. featuring gene products' functional information as part of the graph, may improve predictions. As such, one of the hypotheses behind this work is that by combining the PPI network with a KG composed by Gene Ontology and its protein annotations, the performance of graph mining approaches on PPI networks can be improved.

The protein function prediction problem is commonly deemed as a multi-label classification task, as proteins can have more than one function (Martins et al. (2012)). However, most multi-label classification settings consider each label as a single binary classification case, in which instance the classifier does not take into account any label dependency information when making its predictions. As such, another avenue of research is to elucidate whether it is possible to train a global model for function prediction that leverages the representation of functions along with other sources of protein-related information.

This work relies on the Stanford University's Open Graph Benchmark project (Hu et al. (2020)), which has made available a benchmark dataset on protein function prediction based on a PPI network as well as a set of different ML approaches (including embeddings and graph neural network based) and their results.

To achieve these goals, the following methodology steps are proposed:

1. Investigation and development of different approaches to integrate the Gene Ontology with PPI networks;

2. Evaluation of different machine learning methods coupled with each of the designed approaches to elucidate which ones are better able to capture the ontology knowledge and explore it to improve

predictions.

3. Model evaluation results comparison to benchmark results.

1.2 Contributions

- A novel Graph Neural Network approach that combines PPI and GO representations to learn a general model for protein function that explores background knowledge.
- A methodology to extend existing Graph Neural Network approaches to explore PPIs in tandem with the GO.
- A systematized evaluation of the existing approaches using the OGB benchmark.
- A public release of all software (`https://github.com/lbalbi/PPI_GO_datasets`)
- Three poster presentations:
  - 11th edition of the Bioinformatics Open Days (BOD XI);
  - the 7th LASIGE Workshop ;
  - the 2nd Thematic WideHealth Summer School.

1.3 Document Structure

The present introductory chapter gives a contextualization of the problem underlying the proposed hypothesis and introduces the main objectives and contributions of this dissertation. The remaining five chapters are organized as follows: **Chapter 2** defines and explains the introductory concepts needed to understand the proposed problem; **Chapter 3** surveys several of the most relevant and note-worthy developments in the domain of this work; **Chapter 4** enlightens the two parts of the proposed methodology with a description of its main tasks; **Chapter 5** describes the two approaches at hand: it presents a protein function classification task and uses it as an application of both parts of the proposed methodology; then it further explores the resources used, methods, results, and discussion. **Chapter 6** summarises the main conclusions of this work and considers future developments.

Chapter 2

Concepts

This chapter introduces the principal concepts that provide the background to this work.

2.1 Graphs

Formally, a graph can be defined by a set of objects/entities, i.e, the nodes of the graph, and a set of pairs of nodes, named edges, that indicate the existence of a relationship between the two entities (Latouche and Rossi (2015)). These edges may have a direction, in which case the graph is directed. If the edges do not have an orientation the graph is undirected. Directed edges provide more information than those of undirected graphs. Graphs can also be either homogeneous, where all nodes and edges represent instances and relations of the same type, respectively, or heterogeneous, where nodes and edges are of different types and are labeled accordingly. Graphs can also have weights associated with their edges, making them weighted graphs. These weights take a numerical form and can represent, for example, lengths or connectivity strengths.

Since graphs are data structures that represent connectivity between entities, different types of analysis are supported by assessing which nodes are connected and what type of connections these are (Chami et al. (2020)). While graphs can be easily defined and have relatively simple structure, they are powerful and flexible data structures, with a huge set of applications in several domains of knowledge, including within the biological and Biomedical fields, where they can structure protein-protein interaction (PPI) networks, biochemical networks, transcriptional regulation networks, signal transduction or metabolic networks, and potential applications of graph analysis in such networks are, for example, the determining the role of poorly characterized genes or gene products or identifying drug targets.

2.2 Machine Learning

Machine learning (ML) is a branch of Artificial Intelligence that focuses on the automated learning of data and the study of computer algorithms that allow computer programs to automatically improve their performance at a specific task through experience (Mitchell et al. (1986)). ML is a valuable tool that complements the human understanding of the world, which is built on observing and making hypotheses, deriving predictions from them, and then carrying out experiments to test the validity of said predictions.

When shown examples of data and ground-truth values for the task, ML algorithms can learn the underlying distribution models that best solve the task, in a heuristic sense.

The process of using machine learning (ML) algorithms to build models from data is called learning or training. Since a learned model corresponds to the underlying rules about the data, it is also called a hypothesis, and the actual underlying rules are called the facts or ground-truth. Then, the objective of machine learning is to find or approximate ground-truth (Zhou (2021)).

In machine learning, classification tasks are usually a form of supervised learning. This ML branch is analogous to the human way of learning from past experiences to gain new knowledge that may improve one's ability to perform real-world tasks. Analogously, supervised machine learning learns from labelled data, which represents ”past experiences” in some real-world applications (Jaramillo-Garzón et al. (2016)).

In Bioinformatics, one aims to employ information technology to understand biological observations by obtaining knowledge from pattern discovery. This process involves data collection, retrieval, and data mining. Among these tasks, data mining is where machine learning shines. Data mining is about extracting meaningful patterns from massive datasets to discover knowledge. Researchers have already successfully applied various ML approaches to solve complex bioinformatics problems.

2.2.1 Graph Mining Tasks

As said before, datasets are often adapted into a graph structured representation, in which each node corresponds to a data point, and an edge connects two nodes if the corresponding samples are correlated. As such, many important tasks in network analysis involve predictions over graph elements, such as nodes and edges (Xia et al. (2021)). The data mining of graph-structured data for knowledge is called graph mining.

Graph mining tasks are categorized based on their ability to perform queries as node-level tasks, such as node classification, at an edge-level as link prediction tasks, and at a graph-level, like prediction of triples and graph classification (Parthasarathy et al. (2010)). In link prediction the aim is to discover associations between nodes, so the task is to predict an edge connecting two given nodes or to predict one of the two nodes when given the second node and a labeled edge. In graph classification, for example, the focus is in searching a given graph for patterns that exhibit a specific network structure and using them to classify said graph (Rehman et al. (2012)). The focus of this work will be on solving a node classification problem, with a biological application onto protein function prediction. Node Classification is the method of discovering a representation that demonstrates and distinguishes data classes or concepts, so as to use it to predict a class of objects whose class label is unknown. In a binary classification task nodes may assigned with one of two classes, while in a multi-class setting a node will be labelled with one of several classes. Though there are multiple classes in multi-class classification, each sample belongs to a single class. If more than one label is to be assigned, then it turns into multi-label learning.

2.2.2 Multi-Label Classification

In a Multi-Label Classification problem an instance may be associated with multiple labels, as opposed to the traditional binary or multi-class classification, where each instance is only associated with a single class.

The most general and well-known approach to solving multi-label classification problems is similar to the multi-class's and consists of training an independent classifier to solve a binary problem for each label. The positively-predicted class labels are then assigned to the test instance. However, this method is limited by the fact that dependencies between labels are not explicitly modelled (Ghamrawi and McCallum (2005)). Because they do not exploit dependencies between labels, such techniques are only well-suited to problems in which the classes are independent. However, in many domains labels are actually highly interdependent. In a protein-protein interaction network, for example, one might be interested in predicting functional labels of proteins. Protein function prediction is a multi-class, multi-label problem where each protein may have multiple distinct functions Fa et al. (2018).

2.3 Neural Networks

Deep learning (DL) is a sub-domain of machine learning that allows models made up of multiple processing layers to learn representations of data with multiple levels of abstraction - representation learning-based models –, such as the Neural Networks (NNs) (Schmidhuber (2015)). These Networks can be distinguished from other ML methods due to being able to learn the relationships between complex data features and internally select those features that best represent this knowledge, which is then learnt through the computation of internal parameters and correction of the neural connections' weights. Not only do they resemble the human thinking process by being able to infer from the input data without requiring previous feature extraction, they were actually conceived to resemble the way in which the human brain works.

Overall, the task for the network is to relate the variables it receives in the input layer to some desired behavior at the output layer by repeatedly presenting the examples to the network in a process known as training (Abdi (1994)). Once trained, the neural network should be capable of predicting an output given a previously unseen set of inputs.

Networks are comprised of node layers, containing an input layer, one or more hidden layers, and an output layer (see example in fig. 2.1). These layers consist of many units, the neurons, that are interconnected (Zupan (2017)). Typically, each neuron of one layer connects with all the neurons of the next layer. A neuron takes a number of inputs, either raw inputs or outputs of other neurons, and produces a single output which may then be inputted into the following neuron (Abdi (1994)).

The neuron receives a set of input signals and uses an internal activation function to aggregate its inputs and produce the output signal (Dayhoff (1989)). The internal function can be as simple as a step function, or a more complex transformation such as the commonly used sigmoid (or 'logistic') function. Based on these functions, the activation function makes a 'decision' where if the output of the neuron is above the specified threshold value, that neuron is activated, firing off a signal to the next layer of the network. Otherwise, no signal is passed along to the next layer of the network (Hertz et al. (1991)). The weighted connections propagate signals from unit to unit modifying the strength of the signal according to their weight. Weights are modified in the training process and provide much of the learning capability of the network. Many other types of functions in the NN -related literature, the 'activation functions', exist within neurons to convert incoming values into an output value. The principle behind this neural architecture is that although the functions themselves are very computationally simple, when assembled

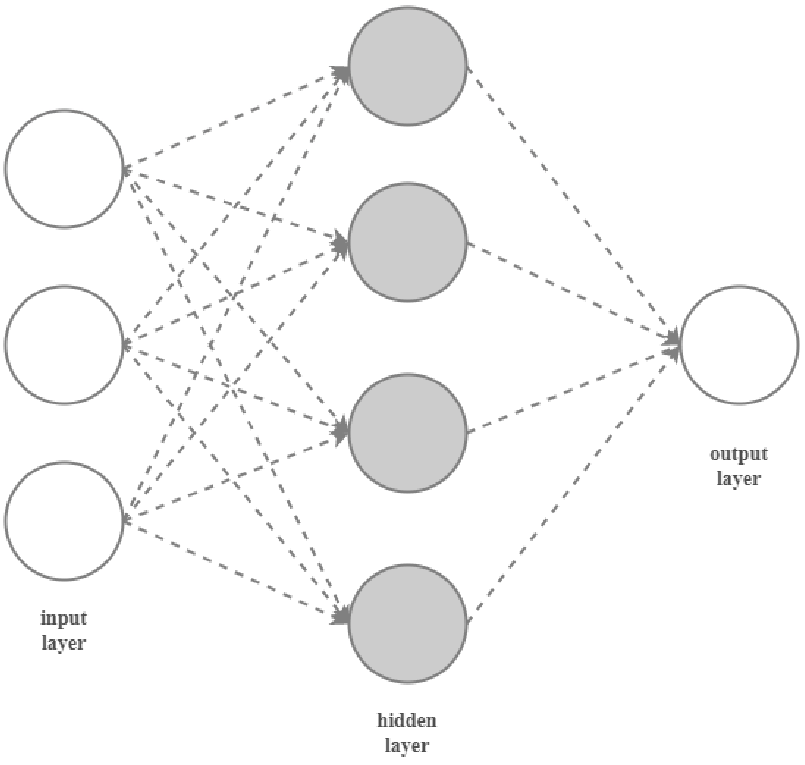


Figure 2.1: Example of an artificial neural network architecture with three layers.

together and connected with weighted connections, they allow for highly complex computation.

The step function ensures an 'all-or-nothing' response is given, whereas the sigmoid function produces a more graduated response where it checks whether an internal threshold is exceeded (Han and Moraga (1995)), and if it is to calculate the output as the function of 1 divided by 1 plus the exponent of the value it has calculated (Hu et al. (2005)). This produces an output that ranges from 0 to 1.

The response of the network is compared to the ground-truth. The discrepancy between the two is calculated and the network then makes changes to its internal weights to reduce the error the next time this input data is presented (Kumar (2017)). Sets of input data and target outputs are repeatedly presented to the network and it adjusts the parameters (connection weights and activation function parameters) to optimize the prediction accuracy. This way, the neural network can infer what input points and data patterns are more important to the understanding of the data and use the patterns learnt to predict the behaviour/classification of unknown data.

Typically, a neural network model's internal weights are updated using the back-propagation of a gradient descent algorithm. The model with a given set of weights is used to make predictions and the error for those predictions is calculated. The back-propagation algorithm navigates down the error gradient by indicating how the network should change its weights to best compute the representation in each layer from the representation in the previous layer (Fahlman (1999)). This is so that there may be a reduction of the prediction error during the next evaluation steps. For the model prediction errors to be calculated and minimized, a loss function is required (Li et al. (2020a)).

Most multi-label classification settings tend to be treated as individual binary classification tasks. As such, among the different losses, the most common in multi-label classification settings is the Binary Cross-Entropy loss (Read and Perez-Cruz (2015)), which measures the performance of a classification model whose output is a probability value between 0 and 1 and computes a loss score that penalizes the prediction based on how far from the actual expected value , of either 0 or 1, it is (Ruby and Yendapalli (2020)), that is then used by the model to adjust its internal parameters.

2.3.1 Simple Feed-Forward Network

The simplest of neural architectures are the Perceptrons (Rosenblatt (1958); Minsky and Papert (1969)), binary classifiers that had only an input layer and an output layer, and were only able to discriminate linear relationships between variables. Rumelhart et al. (1986) proposed the multi-layer perceptron, where hidden layers were introduced to increase the model's predictive power over data with non-linear relationships among its features. This is the widely adopted MLP network design, a simple feed-forward neural network with a low number of layers, usually three or four. Assuming that a given MLP architecture has only one hidden layer of m units and one output unit, the output of the network is given by:

$$y = f(\sum_{k=1}^{m} w_k f_k(\sum_{j=1}^{l} w_{kj} x_j)), \tag{2.1}$$

where fk, k = 1, 2, ..., m, and f denote the activation functions of the hidden-layer neurons and the output neuron, respectively; wk, k = 1, 2, ..., m, and wkj , k = 1,2 ,...., m, j = 1,2 ,..., l denote the weights connected to the output neuron and to the hidden-layer neurons, respectively.

2.4 Graph Neural Networks

Graph neural networks (GNNs) are deep neural networks that operate on domains defined on graph structures, generating representations of nodes that actually depend on the structure of the graph, as well as of node feature information (Chami et al. (2020)). Currently, most graph neural network models have a somewhat universal architecture in common. They resort to a form of neural message passing mechanism in which vector messages are iteratively exchanged between nodes and updated (Gilmer et al. (2017)).

As mentioned above, GNNs use graph structure and node features to learn a representation vector of a node. They do so in their forwarding step, which consists of learning a message passing algorithm and subsequently employing a neighbourhood aggregation strategy to iteratively update node beliefs and down-stream compute a function of their entire input graph.

The message-passing mechanism is analogous to a multi-layer perceptron, as it mainly relies on linear operations usually followed by a single element-wise non-linearity operation. In the message passing scheme, each node aggregates the messages (feature vectors) incoming from its neighbours to compute its new feature vector (Xu et al. (2018)). The neighboring nodes' messages, in turn, are based on information aggregated from their respective neighbourhoods, and so on (see fig. 2.3). Afterwards, the node feature representation is then updated by being combined with the vector representations of the node's neighbourhood through a linear combination operation; and finally, an element-wise non-linearity function is applied (Wu et al. (2020)).

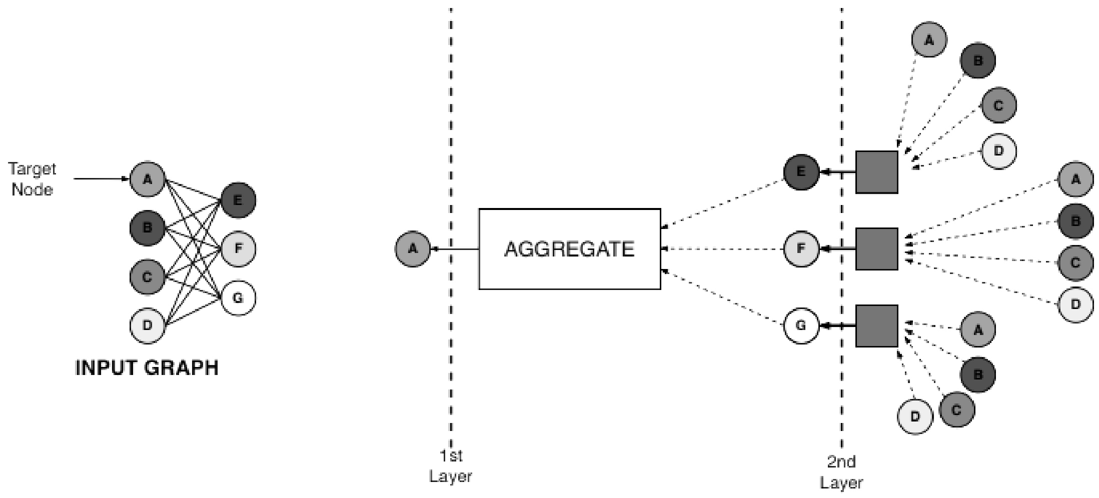


Figure 2.2: Example of a graph neural network's message-passing mechanism.

Message passing is repeated throughout multiple rounds, and as the iterations progress each node embedding contains more and more information from further reaches of its neighbourhood.

The basic GNN message-passing function is defined as is seen in equation 2.2:

$$h_u^{(k)} = \sigma(w_{self}^{(k)} h_u^{(k-1)} + w_{neigh}^{(k)} \sum_{v \epsilon N(u)} h_v^{(k-1)} + b^{(k)}) \qquad , \tag{2.2}$$

to determine the feature representation h of node u at a layer k, where σ is the activation function, w is the weight matrix, $h_u^k - 1$ is the node's initial feature representation or its belief at the previous layer k-1, $h_v^k - 1$ is neighbouring node v's feature representation at layer k-1 and b is the bias associated to layer k.

A limitation of the basic GNN architecture is how difficult it is for the node features to affect the node's state after multiple updates/iterations, since structural information prevails in the nodes' feature representations.

2.4.1 Graph Convolutional Neural Networks

CNNs leverage feature pooling to extract localized spatial features and shared weights. This pooling approach that allows for a graph-level representation for each node is crucial to solving graph data mining problems (LeCun et al. (2015); Albawi et al. (2017)). However, these networks only operate on regular Euclidean data. As such, several graph neural network variants based on CNNs and existing graph embedding approaches were proposed.

Possibly the most famous development of this kind of network are the Graph Convolutional Networks (GCN), proposed by Kipf and Welling (2016). GCNs can use a set of different aggregator functions that allow it to represent a node by aggregating feature vectors of neighboring nodes, analogously to how the convolutional kernel works in Convolutional Neural Networks (CNNs), by means of a pooling function, which may either compute the sum, the median, or a normalization of the set of node embeddings.

While having advanced the state-of-the-art in graph representation learning, the GCN does have shortcomings, being one of them the fact that it considers all of the nodes' neighbours when learning node representations. This results in discrepancies in the amount of neighbourhood information being aggregated to the nodes feature representations. Nodes with scarce neighbourhoods will mainly depend on their own feature representation while nodes with many neighbours will have representations that mainly reflect node connectivity and almost do not take into account the node's own features.

One of the most well-known architectural developments that address this neighbourhood limitation is the GraphSAGE (Hamilton et al. (2017)), that randomly samples a fixed-size set of neighbours for each node.

2.5 Graph Representation learning

Traditional machine learning algorithms cannot work directly on graph-based data, as they require numerical vectors as input. As such, an approach is to learn feature representations by solving an optimization problem. Graph Representation Learning captures representations of graph components such as nodes, the edges between them, or sub-graphs, through dense low-dimensional numerical vectors (Barros et al. (2021)).

In node-level representation learning in particular, embedding approaches use dimensionality reduction techniques to distil the high-dimensional information about a node's graph neighbourhood into a dense vector embedding while preserving the inherent relational properties in an embedding space. Some of the most popular feature extraction methods are node degree, centrality methods, or clustering coefficients. With graph-level features, a more general approach is taken, where the full graph structure is represented. These embeddings can then be fed to downstream machine learning systems and aid in tasks such as node classification, clustering, and link prediction (Hamilton (2020)).

As such, in this section several unsupervised GRL techniques that can be applied to graph analysis will be introduced. These methods take a graph as input and produce a latent representation as an output. Such techniques can be categorised into:

i) Factorization-based approaches, that represent the connections between nodes in the form of a matrix and mainly focus on matrix factorization approaches to obtain node embeddings, e.g. Laplacian Eigenmaps (LE) (Belkin and Niyogi (2001)), Graph Factorization (GF) (Ahmed et al. (2013)), GraRep (Cao et al. (2015)) and HOPE (Doosti et al. (2020));

ii) Random walk-based approaches, that resort to simulating random walks over graphs to capture the structural relationships between nodes, such as node2vec (Grover and Leskovec (2016)), a development of DeepWalk (Perozzi et al. (2014)) that uses biased-random walks to traverse a node's neighbourhood in an undirected fashion, and uses SkipGram (Mikolov et al. (2013)) to maximize the co-occurrence probability of the neighbours conditioned on a given node embedding. Node2vec's biased walks are controlled by two hyperparameters, p and q, that regulate the likelihood of the walks immediately revisiting the source node and the likelihood of immediately revisiting the node's one-hop neighborhood, respectively;

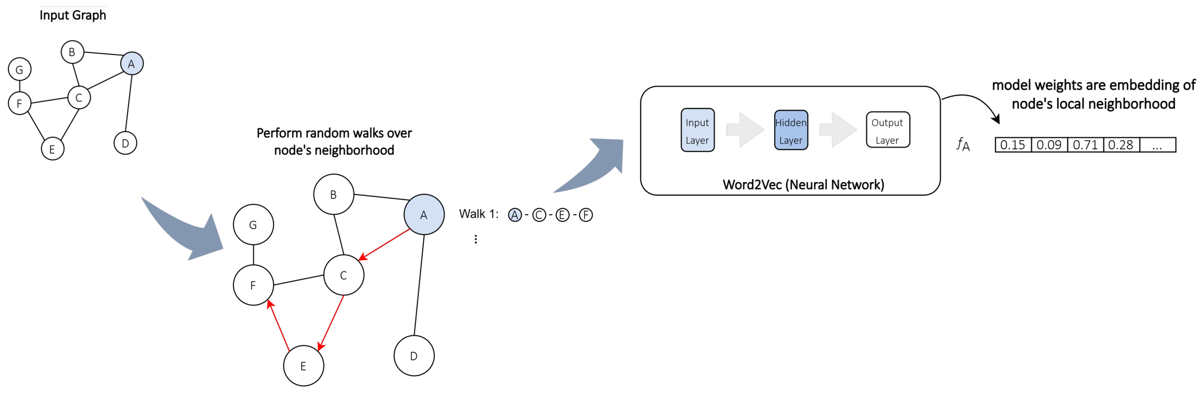

Input Graph
Perform random walks over node's neighborhood
Walk 1: A - C - E - F
Input Layer
Hidden Layer
Output Layer
Word2Vec (Neural Network)
model weights are embedding of node's local neighborhood
f_A
0.15
0.09
0.71
0.28
...

Figure 2.3: Example of a random-walk -based embedding mechanism.

iii) Neural network-based approaches, that leverage the use of auto-encoders that aggregate and incorporate structural information about a node's local neighborhood directly into the encoding algorithm (Hamilton (2020)), which is not possible for the other representation-learning approaches (Cao (2016)). Graph convolutional neural networks, in particular, are able to obtain node representations by aggregating both connectivity information and node attributes for a node's local neighbourhood, as explained in Section 2.4.1.

2.6 Ontologies

The biological domain differs from other scientific fields in the sense that it is not a rigidly defined domain that can be completely modelled with mathematics. Its knowledge needs to be captured and represented in a different form. Ontologies are formal representations of a specific knowledge domain, that formally defines the set of concepts and categories that belong to and make up the knowledge of a given domain alongside their properties and relations among them (Uschold and Grüninger (1996); Guarino (1997)). This is a means of classifying concepts via a hierarchical structure of classes. Ontologies can only contain facts that are considered to be true for all entities of a particular type. For that reason, they do not represent entities but instead classes. The formal structure of ontologies allows for the usage of the domain knowledge it represents in data mining tasks.

The massive amounts of data produced by modern day research in genomics require explicitly defined vocabularies to describe the attributes of biological entities (e.g. genes and gene products) in order to allow their integration, organization and data reasoning. The assignment of real-world entities to their semantic description in a given domain is what a semantic annotation consists in (Kiryakov et al. (2003)). Relying on semantic annotation of biological entities allows automatic reasoning to be applied directly to them.

The Gene Ontology (GO) project (Ashburner et al. (2000)) is one of the most successful systematic descriptions of current biological knowledge. It aims to cover the whole universe of protein functions by maintaining a dynamic, structured, precisely defined, and controlled vocabulary of classes for expressing the roles of genes and gene products (Bard and Rhee (2004)). The reason GO is considered as dynamic is because its structure is ever changing, being updated as more information is available.

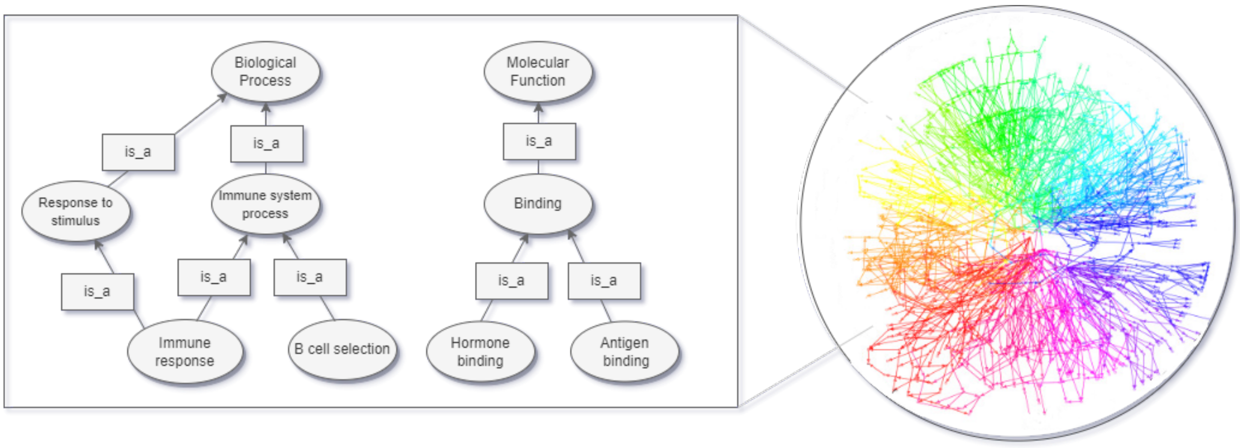
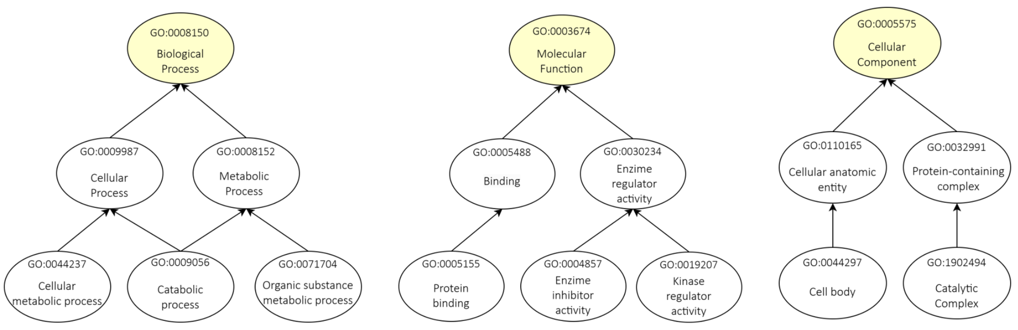


Figure 2.4: Partial depiction of within-domain GO structural hierarchy; Biological Process and Molecular Function are both root classes of their own sub-ontologies.

The GO ontology is represented as a directed acyclic graph (DAG) in which the GO classes are nodes that are linked to each other through edges that represent “is a” and “part of” relationships, among others (Egaña et al. (2007)). This kind of organization allows for the structuring of biological knowledge as a graph-like representation. GO classes are organized hierarchically, in a tree-like structure, with the most general classes at the root of the GO tree and the most specific at the leaves, distributed across three major semantic domains – molecular function, biological process, and cellular location. The more general classes are named ”parent classes”, while the more specific ones are named ”children classes” and can have multiple parents.



GO:0008150 Biological Process
GO:0009987 Cellular Process
GO:0008152 Metabolic Process
GO:0044237 Cellular metabolic process
GO:0009056 Catabolic process
GO:0071704 Organic substance metabolic process
GO:0003674 Molecular Function
GO:0005488 Binding
GO:0030234 Enzime regulator activity
GO:0005155 Protein binding
GO:0004857 Enzime inhibitor activity
GO:0019207 Kinase regulator activity
GO:0005575 Cellular Component
GO:0110165 Cellular anatomic entity
GO:0032991 Protein-containing complex
GO:0044297 Cell body
GO:1902494 Catalytic Complex

Figure 2.5: Dual-input neural network architecture. The GCN model used in each neural branch can either be the one proposed by Kipf and Welling (2016) or GraphSAGE.

Besides the defined classes and the structured relationships between them, the GO ontology also contains the associations between gene products and the classes (Yon Rhee et al. (2008)). As such, when a gene is annotated to a GO class, associations between the gene and the classes’ parents are implicitly inferred, since annotations to a GO class inherit all the properties of its ancestors.

From the perspective of modern functional analysis, the GO provides a universal and standardized way to assess whether a set of proteins have similar functions, which has led to it being increasingly resorted to in function prediction procedures used in model organism settings (Lin et al. (2004b)).

2.7 Knowledge Graphs

Ontologies and linked data are usually represented by directed multi-relational graphs designated as knowledge graphs (KGs). A KG is an abstraction used in knowledge representation to encode knowledge in one or more domains by representing entities (i.e., nodes) and the binary relationships that connect them (Dimou (2020)). Its nodes represent entities or entity properties and labeled edges represent specific relationships between them (Bellomarini et al. (2020)).

KGs can be very valuable to data mining tasks, as they can provide background information in a more graph-like representation (MacLean (2021)). Using these KGs to enhance traditional data mining techniques with background knowledge is a relatively recent endeavor, since most ML methods are unable to process KGs directly. However, as mentioned before, there are techniques that can be applied to learn vector representations for each of the entities inside a graph based on a limited set of global latent features (Geleta et al. (2021)). Most of these methods can then be used for different downstream tasks.

2.7.1 Biological Knowledge Graphs

Cellular functions and biological systems are carried out through a complex network of interactions between biomolecules. The topology and dynamics of these complex biological networks can be readily studied by graph theory. The nature of these networks is closely associated with the topology and the biological functions of the nodes and edges. By modeling these biological networks into biological knowledge graphs, the biological relations between entities and connections between entities and the meaningful background knowledge will be represented within a universal formal structure (MacLean (2021)).

Chapter 3

Related Work

This chapter makes a brief introduction on the relevant advancements made in the main fields that encompass the work developed for this thesis.

3.1 Protein Function Prediction with Machine Learning

Most machine Learning approaches available for the prediction of protein functions explore how protein functionality is intimately linked to several biological aspects that make up their characterization, such as structural and evolutionary features they may have and the network of interactions they are inserted in (Sarkar and Saha (2019)). Bonetta and Valentino (2020) explore protein structure information, by mining for similar structural motifs and domains. King et al. (2000) leverage protein amino acid sequence similarity to infer functional similarity among proteins. PPI network -based approaches, however, will be the focus of this work. As shown by Zhang et al. (2009), these approaches base their predictions on the exploring of query proteins' physical interactions with other proteins whose functions are known.

3.2 Exploring ontologies and PPIs

Several works have resorted to exploiting GO information to improve predictive power in different tasks on PPI networks, including PPI prediction and protein function prediction. Among these, various studies have recognized similarity in GO annotations as one of the strongest predictors of PPIs (Lin et al. (2004a);Patil and Nakamura (2005)). GO annotation-driven interaction inference is based on the observation that proteins who share the same cellular compartment or a common biological process or molecular function are more likely to interact. The literature distinguishes at least two different approaches to exploit GO annotations for the prediction of protein interactions: the semantic similarity measure (SSM) approach; and the machine learning (ML) approach.

SSM methods operate on hierarchical graphs of relationships such as GO to measure similarities between entities. As such, GO classes who are closer to each other in the GO Graph are expected to have higher similarity. Sousa et al. (2020a) proposed evoKGsim, an approach where a set of semantic similarity (SS) measures is computed over a knowledge graph that included the GO and protein annotations to it and used the SS scores as features on a Genetic Programming method that attempts to make protein-protein interaction predictions.

When employing ML approaches for PPI inference, the traditional approach is to encode shared GO classes between two proteins within a feature vector, which is subsequently evaluated by an ML classifier. This kind of approach ignores the relationships between GO classes and therefore does not exploit important topological information from the GO.

Most recently there's been a trend to incorporate some GO information to learn label dependency for protein prediction problems by using hierarchical neural networks. DeepGO (Kulmanov et al. (2018)) is a deep learning-based approach that uses both PPI network and protein sequence features to predict protein function and uses a hierarchical neural network to learn label dependencies from GO structural information. Sureyya Rifaioglu et al. (2019) proposed a hierarchical multi-task deep neural network architecture with as many parallel hidden layers as class labels and that learns GO classes' dependencies based on their GO category's hierarchy to make protein function prediction on PPI networks.

With the creation of ML models that are able to reason over graph data, the exploitation of GO information on PPI network prediction tasks may be reaching a turning point, as both data and knowledge can now be explored by the models without needing to be encoded into dense numerical representations for them to learn from its topology.

3.3 Open Graph Benchmark

The Open Graph Benchmark (OGB) (Hu et al. (2020)) is a community-driven initiative for the collection of realistic, large-scale, and diverse benchmark datasets of varying sizes that cover different graph machine learning tasks. Their goal is to facilitate scalable, robust, and reproducible graph machine learning (ML) research over a diverse range of domains, ranging from social and information networks to biological networks. They currently provide 5 datasets adopted from 3 different application domains for predicting the properties of individual nodes. The OGBn-proteins dataset is a protein-protein association network. Nodes represent proteins, and edges indicate different types of biologically meaningful associations between proteins, e.g., physical interactions, co-expression or homology (Szklarczyk et al. (2019)). This dataset allows for model evaluation on a protein function prediction task.

Open Graph Benchmark:
Datasets for Machine Learning on Graphs

Weihua Hu[1], Matthias Fey[2], Marinka Zitnik[3], Yuxiao Dong[4],
Hongyu Ren[1], Bowen Liu[5], Michele Catasta[1], Jure Leskovec[1]
[1]Department of Computer Science, [5]Chemistry, Stanford University
[2]Department of Computer Science, TU Dortmund University
[3]Department of Biomedical Informatics, Harvard University
[4]Microsoft Research, Redmond
ogb@cs.stanford.edu

Steering Committee
Regina Barzilay, Peter Battaglia, Yoshua Bengio, Michael Bronstein,
Stephan Günnemann, Will Hamilton, Tommi Jaakkola, Stefanie Jegelka,
Maximilian Nickel, Chris Re, Le Song, Jian Tang, Max Welling, Rich Zemel

Figure 3.1: OGB Project's paper on benchmark collection and evaluation of various DL state-of-the-art models.

Furthermore, OGB provides a common codebase using popular frameworks for loading, constructing, and representing graphs as well as code implementations of established performance metrics for fast model evaluation and comparison. They evaluated the following representative neural-based models over the protein function prediction task: A multilayer perceptron (MLP) predictor that receives PPI confidence mean scores as node features, having no graph structure information as input. This predictor can also jointly receive the PPI scores with node2vec embeddings of the PPI network (Grover and Leskovec (2016)); a Graph Convolutional Network (Kipf and Welling (2016)) that encodes the topology of the network into several node embeddings and uses it to update its node beliefs; and a GraphSAGE network (Hamilton et al. (2017)), that works similarly to the GCN, adopting a mean pooling variant and a simple skip connection to preserve central node features.

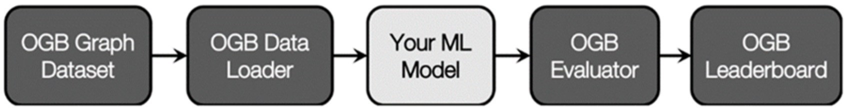


Figure 3.2: Outline of OGB's public contributions, from benchmark datasets to ML frameworks with domain-relevant applications. Retrieved from the OGB's github page.

As such, in addition to building the datasets, they also performed benchmark experiments for each dataset, the results of which they made public in their leaderboards, which they use to keep track of the state-of-the-art.

Since then, the Gated Attention Network (Zhang et al. (2018)), an aggregator-based architecture, has managed to outperform the baselines. Similarly to its antecessor, Graph Attention Network (GAT) (Velickovic et al. (2018)), the model incorporates an attention mechanism into the propagation step, by attending to a given node's neighbours and assigning them with different weights to retain only the most meaningful sub-neighbourhoods and achieve better results.

Other developments to model evaluation over the benchmark dataset have included testing with DeepGCN and DeeperGCN, two implementations from the same team, Li et al. (2020b), that fundamentally allow for the usage of higher numbers of hidden layers in GCNs without suffering from vanishing gradient issues. Kong et al. (2020) introduced FLAG, a model that applies augmentation operations on the node feature space while preserving the graph structure.

These are some of the works made in the aftermath of opening the OGB leaderboard. Its results progress with the advancements made in increasing model complexity and computational power. A common factor among the leaderboard's additions is that they mostly focus on developing approaches that use highly parameterized neural models, but close to none explore ways to improve predictions by enriching data by adding contextual knowledge.

Chapter 4

Methodology

The proposed methodology for this work is divided into two branches that explore different approaches into incorporating background functional knowledge into protein function prediction based on PPIs.

The first comprehends the development of several neural-based approaches where prediction is made over a holistic graph that combines the PPI graph with a knowledge graph composed of the GO graph and its annotations for the proteins in the PPI graph. These approaches allow predictors to explore information about a protein's functions, and how these functions are related to each other, in tandem with its interactions with other proteins.

The second methodology sees the construction of a dual-graph neural network -based approach where two different protein representations are learned in parallel: one based on protein interactions and another based on known protein functional aspects.

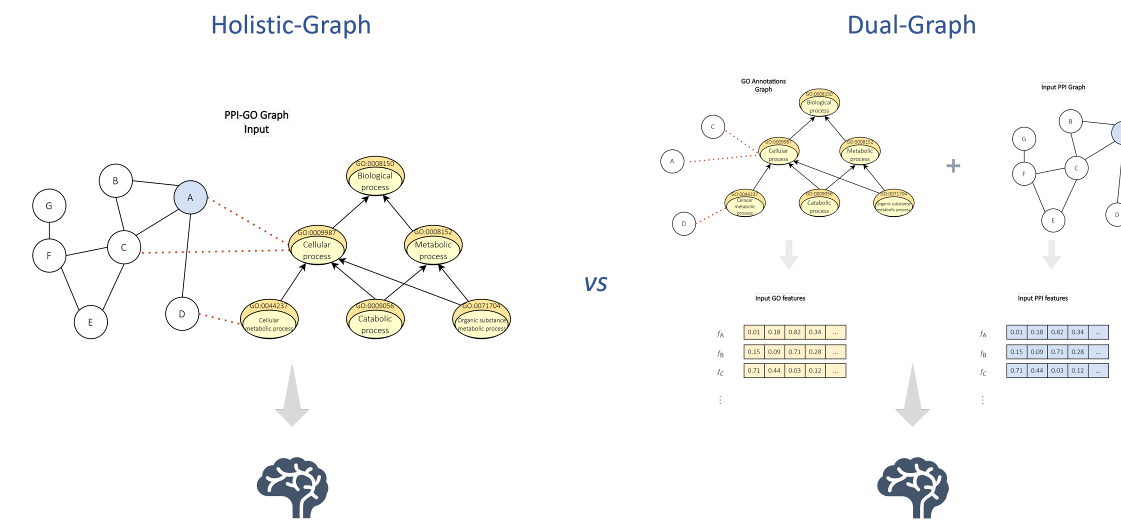

Holistic-Graph
PPI-GO Graph Input
A
B
C
D
E
F
G
GO:0008150 Biological process
GO:0009987 Cellular process
GO:0008152 Metabolic process
GO:0044237 Cellular metabolic process
GO:0009056 Catabolic process
GO:0071704 Organic substance metabolic process
vs
Dual-Graph
GO Annotations Graph
Input PPI Graph
+
Input GO features
Input PPI features
f_A 0.01 0.18 0.82 0.34 ...
f_B 0.15 0.09 0.71 0.28 ...
f_C 0.71 0.44 0.03 0.12 ...

Figure 4.1: Overview of Holistic and Dual-Graph approaches.

This section first describes the common data sources used for this work and then delves into each step of the two proposed methodologies.

4.1 Data

4.1.1 OGBn-proteins

The benchmark PPI dataset was retrieved from the OGB Project's online repository. In general, the search for practically effective machine learning models in a given domain proceeds through a sequence of increasingly realistic and interesting benchmarks. The focus of this work was on the OGB Project's benchmark PPI Network dataset (OGBn-proteins), with a total of 132,534 proteins of eight different species. This dataset covers a multi-class node classification task applied to protein function prediction. This dataset has already been applied by several others in evaluating protein function prediction approaches.

The OGB team split the dataset's proteins into training/validation/testing partitions by species, with a partitioning distribution of 65%/16%/19%, respectively. This work adopts the same distribution scheme to make results comparable to the OGB's benchmark. This by-species splitting means that testing proteins are from a species completely unknown to the predictor. This partitioning enables a *cross-species* evaluation. Thus, there will be an across-species generalization performance assessment of the model.

Another partitioning scheme was made through random protein splitting, while keeping the training/validation/testing relative distribution (%) of the original OGB-made partitions. This allows for a *global*, multi-species classification, where the predictor already has prior knowledge of the species of the proteins it is going to classify during the testing phase. As such, evaluations with this type of partitioning usually produces better results.

The OGBn-proteins dataset contains a "ground-truth" matrix with 1,689,573 positive protein annotations to the 112 GO classes that function as labels for the multi-label classification task. The goal of the multi-label classification task is to correctly predict these positive annotations. From this full set of positive annotations to the entirety of the dataset's proteins, two sets were built with only training proteins' annotations, for both the original by-species partitioning (1,227,366 annotations) and the random protein splitting partitioning (1,134,424 annotations) schemes.

4.1.2 Gene Ontology

The benchmark dataset's classification labels were selected by the OGB Team from the 2020-06-01 version of Gene Ontology (GO)[1]. Therefore, to allow for a complete compatibility between the data and the knowledge this work aims to link it with, the exact same versions of the Full GO and the GO slim Generic sub-ontology were retrieved from the Gene Ontology Consortium's website. The GO data was processed with RDFLib parsing package to filter out obsolete GO-classes and retain meaningful associations, such as "IS_A" and "PART_OF" relationships, between the functional classes. This produced a "clean" full ontology with a total of 44,266 classes and of 71,515 links between them, and a GO Slim Generic with 145 classes and of 144 links. Each of these ontologies can be loaded into a graph where their classes correspond to the graph nodes and the GO links are the edges that connect them. Reminding

[1]based on personal communication

that the GO graph is directed and acyclic (DAG), the links between classes only have one direction, from a source node, in this case a more specific, “child” node, to a more generic “parent” node.

4.1.3 STRING protein interactions and GO annotations

The PPI data used to form the PPI network had been retrieved from the STRING database (v.11.0). The PPIs for the proteins used were predicted by STRING on the basis of all publicly available evidence sources of protein-protein associations such as text mining, experimental evidence, databases, co-expression, neighbourhood, gene fusion and co-occurrence. In general, certain associations may indicate that proteins jointly contribute to a shared function but this does not necessarily imply they have direct binding. To achieve a comprehensive network, STRING complements this information with computational predictions, by annotating each protein-protein interaction with one or more ’scores’, which are defined as indicators of confidence, but not necessarily the strength of the interaction. All scores range from 0 to 1, where 1 is the highest possible confidence. The 132,534 proteins used in this study belong to the 8 different species named before and, according to the benchmark dataset, participate in 39,561,252 interactions with various confidence scores.

We further retrieved from the STRING database functional information pertaining to the dataset’s proteins and filtered it to only retain protein annotations to the Gene Ontology and further on to not include any annotations to the classification task’s labels. A total of 4,567,394 annotations to the entirety of the dataset’s proteins were collected, making up the first set of annotations i) used in the experiments.

Further filtering approaches led to the creation of three other annotation sets:

ii) a second set was obtained by only retaining from the first set annotations to the training set’s proteins, both for a cross-species validation (3,154,433 annotations) and a multi-species validation setting (3,063,260 annotations);

iii) a third set with only non-redundant annotations, with a total of 1,291,670 protein annotations. The non-redundancy exclusion process consists in only keeping direct annotations to the more specific GO classes when faced with two annotations from the same protein to two GO classes where one is a descendant of the other in the GO hierarchy;

iv) the final set pertains to annotations to classes of the GO slim Generic ontology, that were retrieved while taking into account non-redundant annotations to the GO slim classes and annotations that were reachable to the GO slim’s classes through any children classes they may have on the full GO. In other words, if a given GO Slim class has a child class in the full GO to which a given protein is annotated, then the protein is assumed as annotated to the parent class. This produced a total of 491,723 annotations to the GO Slim.

4.2 Holistic Graph -based Protein Function Prediction

The first proposed approach creates a holistic graph PPIs and the GO, then uses it as input for multi-label classification to predict protein function employing several machine learning approaches based on neural networks, graph embeddings and graph neural networks.

4.2.1 Holistic Graph Construction

The holistic approach comprehends the construction of a super-graph that integrates the PPI network and the GO tree. For that, the two graphs are linked together by two types of annotations: the set of collected positive annotations between training proteins and the 112 GO classes that are part of the task's labels (set selection explained in Section 4.1.1); and a set of annotations between the graph's proteins and classes of the GO that are not labels. The latter set was constructed through the use of several approaches to combining the PPI data and GO with the inclusion of different kinds of GO protein functional information: i) (*PPI-GO full*) approach uses the full set of protein annotations to the GO graph; ii) (*PPI-GO train*) uses only annotations to the training set's proteins; iii) (*PPI-GO direct*) uses only non-redundant protein annotations; iv) (*PPI-GO slim*) uses annotations to a GO sub-graph pertaining to the GO Slim Generic.

The four variants of graphs produced are described in Tables 4.1 and 4.2 for both randomly and by-species protein distribution for training, validation and testing partitions, along with their edge selection criteria and respective edge numbers. The sets of annotations to non-label GO classes utilized in this step are those described in Section 4.1.3 along with respective annotation numbers. The graphs produced only contain undirected edges for both between-protein interactions and for links between GO classes and annotations.

Experiences with the *PPI-GO full* graph allow the studying of how the classifier leverages the knowledge of already known functions for the testing proteins it predicts unknown functions for; while with the *PPI-GO train* graph, that does not include testing protein annotations, one can assess how the model behaves when making functional predictions for proteins it does not have any previous functional* knowledge of.

The two remaining graphs make up a secondary branch of experiments with which a study is made on which annotations to the different levels of the GO hierarchy the predictor leverages to improve protein function predictions. With the *PPI-GO direct* graph the aim is to understand if with the predictor learning of the more specific protein functions for both training and testing proteins it will produce better predictions. The intent of removing indirect annotations to broader classes is to potentially remove less informative direct paths to parent classes to which there are already indirect annotations and, with this, reduce noise. The *PPI-GO slim* graph includes solely annotations to a generic GO subset that contains relevant and mostly high-level GO classes. Experiments over this graph aim to test model performance when learning of more general protein functions instead of more specific ones.

Node features

Two types of features were used for graph nodes that represent proteins: PPI-based features and GO-based features.

PPI features: The PPI network of the ogbn-proteins dataset does not have input node features. It has edge features associated to all of its edges. In particular, the features that are included in the benchmark

Dataset name	Edge types	Number of edges
PPI-GO full	PPI edges + GO links + all annotations	45,427,527
PPI-GO train	PPI edges + GO links + train annotations	44,014,566
PPI-GO direct	PPI edges + GO links + non-redundant annotations	42,151,803
PPI-GO slim	PPI edges + GO slim links + GO slim annotations	41,351,856

Table 4.1: Summary of PPI-GO graphs built for a protein function prediction task by species (for posterior cross-species model evaluation as performed by the OGB team), with the edge selection approaches and number of edges that make up each of them. The ”Edge types” column describes the selection process for the sets of annotations to GO classes that are not task labels. The ”Number of edges” includes the numbers of PPI edges, links between GO classes, the set of training proteins’ annotations for the protein splitting-by-species scheme and each experiment’s corresponding set of annotations to non-labels.

Dataset name	Edge types	Number of edges
PPI-GO full	PPI edges + GO links + all annotations	45,334,585
PPI-GO train	PPI edges + GO links + train annotations	43,830,451
PPI-GO direct	PPI edges + GO links + non-redundant annotations	42,058,861
PPI-GO slim	PPI edges + GO slim links + GO slim annotations	41,258,914

Table 4.2: Summary of PPI-GO graphs built for a global protein function prediction task, with the edge selection approaches and number of edges that make up each of them. The ”Edge types” columns describes the selection process for the sets of annotations to GO classes that are not task labels. The ”Number of edges” column includes the numbers of PPI edges, links between GO classes, the set of training proteins’ annotations for the random splitting scheme and each experiment’s corresponding set of annotations to non-labels.

PPI dataset pertain to eight scores, each representing the level of confidence with which a given edge can be classified as a type of interaction. In the baseline experiments, the OGB team used the average edge features of incoming edges as node features, also used in this work.

GO features: The Information Content (IC) measure conveys how informative a given concept is. The ICseco measure is an ontology structure-based information content approach proposed by Hanif and Aono (2010) and is given by:

$$ICseco(t) = 1 - \frac{\log[hypo(t) + 1])}{\log(maxnodes)} \tag{4.1}$$

where hypo(t) is the number of direct and indirect descendants from class t (including class t) and maxnodes is the total number of concepts in the ontology. The further down the ontological hierarchy a class is, the more specific it is, therefore, the more informative it will be.For each of the knowledge graphs built, the ICSeco measure of the GO classes was determined. Then, for each of the network’s proteins, an average of the ICSeco score of the GO classes they are annotated with was computed and used as a protein node feature.

4.2.2 Machine Learning Architectures

For the multi-label classification task, three different neural algorithms were used: MLP, GCN and GraphSAGE. These models consist of three-layered (one hidden layer) architectures implemented by the OGB Team on PyTorch (this work was done with v.1.10.2). The final output represents a prediction of if the node belongs to any of the classification labels and is then fed onto a Binary Cross Entropy with

Logits Loss function (BCEWithLogitsLoss). A learning rate of 0,01 was used for all neural networks.

The Multi-Layer Perceptron (MLP) takes node features as input, and passes them through several linear layers, in between each it applies a ReLu function to the output. It is a simple network that does not take into account graph structure. As such, one of the ML approaches used saw the node2vec embedding method being used upstream for node feature extraction over the graphs, with its embeddings being used as input node features for the MLP network, either solely or concatenated with other node features. The OGB team's also PyTorch-based node2vec implementation was used with default parameters, along with a random walk length of 40 nodes and a maximum of 10 walks per node. The embeddings generated with this method are 128-dimensional feature vectors that can be general representations of proteins in the network and are used as feature vectors for the protein function prediction task.

The Graph Convolutional Network (GCN) and the GraphSAGE receive both node features and an adjacency NxN matrix, where N is the number of nodes. This adjacency matrix represents the edges between nodes, this way capturing the graph's structural information. Prior to being given to the GCN, the matrix suffers a few normalization operations, e.g. a Laplacian normalization process, done with PyTorch Geometric.

4.2.3 Evaluation

The proposed task for the multi-label classification setting is to predict the presence of protein functions out of 112 functional labels to predict in total. As such, every node can be assigned one or more labels.

This work's baselines were established by replicating the evaluation of the OGB's ML approaches over their benchmark PPI dataset for the two splitting scenarios in our machine (NVIDIA GeForce RTX 2080 Ti GPU) with PyTorch v1.10.2 for Cuda 10.2, so as to produce results that could be directly comparable to those of our experiments.

The three OGB-implemented DL algorithms described in section 4.2.2 were combined with each of the experiments proposed in section 4.2.1. These couplings were then submitted to evaluations using the random and species-based splitting scenarios presented in 4.2.1, as seen in Table 4.3.

Three alternatives for node features were evaluated: a) PPI features: 8-dimensional mean values of String PPI scores; b) PPI features + GO feature; c) None. The GCN and GraphSAGE implementations require an initial embedding implementation. As such, the employment of c) was made by substituting the 8-dimensional PPI scores for 8-dimensional vectors with a single score of 0.5 for all nodes. The MLP only makes predictions based on node-features, so it was not possible to apply c) while using this method unassisted. The MLP+node2vec implementation allows for the substitution of the PPI score features by the node2vec node embeddings. Therefore, node feature approach c) was solely employed with three of the four proposed ML implementations.

Before being fed to the model, the graphs' edges that pertain to incoming links to a given protein

GO graph	Annotation sets	Node Features	Dataset Partitioning
Full GO	all annotations	8 PPI scores	Per species
	w/o test proteins	8 PPI scores + IC_{Seco}	
GO Slim Generic	non-redundant	8 PPI scores + node2vec	Random
	GO Slim Generic	8 PPI scores + IC_{Seco} + node2vec	

Table 4.3: Experimental conditions tested for the protein function prediction problem.

are duplicated in the opposite direction, to reflect undirected links between proteins. This was also done for edges that represent annotations between proteins and GO classes. The edges that represent links between classes were not duplicated as they should reflect the hierarchical relations inside the GO tree.

Training was performed using an Adam Optimizer (Kingma and Ba (2015)) and for 1000 epochs, using the default model hyper-parameters established by the OGB Team. At every 5 epochs the model had an evaluation step. Two evaluation metrics were selected for being very common for evaluating classification models in biological contexts, specifically in multi-class/multi-label prediction settings, which are described below:

1. Area under the Receiver Operating Characteristic Curve (ROC-AUC) – The ROC is the curve resulting from plotting the false positive rate $\frac{TP}{TP+FN}$ against the true positive rate $\frac{FP}{TN+FP}$. The AUC, or area under the ROC Curve, gives the probability that a classifier will rank a randomly chosen positive example higher than a randomly chosen negative one. It measures how much the model is capable of distinguishing between the two classes.

2. Accuracy - It is the proportion of examples correctly classified. The overall accuracy is the ratio of the total number of correctly predicted samples over the total number of samples. It is not a very informative metric and in scenarios where there is a high imbalance of classes (such as in this work's case) it can be very omissive of the model's real predictive power.

The best test score metrics were selected according to the evaluation epoch in which the validation ROC-AUC scores were the highest.

4.3 Dual-Graph -based Protein Function Prediction

While the focus of the first part of this work was to develop approaches to directly integrate the GO with a PPI network and validate its capacity to increase predictive power of unknown protein functions, the second part of the work explores the parallel learning of protein representations based on two distinct graphs, one that represents physical interactions between proteins and another that includes the dataset proteins' annotations in the GO graph.

4.3.1 Dual Graph Construction

This methodology proposed the use of two distinct graphs, one representing the PPI network of the prediction proteins and the other contains protein annotations to the GO.

For the PPI graph, the benchmark dataset's PPI network is used. For the protein GO annotations' graph, a first *GO full* graph for each of the protein partition schemes was produced, built by including both training and testing protein non-redundant annotations to non-label GO classes and by including the set of training proteins' annotations to the functional labels to predict. The sets of annotations utilized in this step are the same as the sets used in the methodology of the Holistic Graph approach, with annotations' numbers described in Section 4.1.3.

From these initial graphs six more graphs (three graphs for each partitioning scheme) were created, with the intent of exploring what type of annotations will be informative enough to provide a protein function representation that the model will be able to learn from to improve predictions: 1) two *GO train* graphs were obtained by including only annotations from training proteins as links to the GO components for each partitioning; 2) the remaining four graphs were obtained by including annotations from either one of the two sub-ontologies to which the dataset's label GO classes belong to, Molecular Function (*GO MF Pred*) or Biological Process (*GO BP Pred*), for both OGB-based and random splitting.

The GO graphs produced contain directed edges between GO classes to reflect GO hierarchy, while the PPI graph contains undirected edges between proteins and between proteins.

The experiments performed over the *GO full* graph explore how testing proteins' functional representations containing said proteins' already known functional information may influence the neural approach's predictions of their unknown functions. Experiments over the *GO train* graph assess the model's lack of previous knowledge of testing proteins' known functions when making predictions over their unknown functions.

The *GO BP Pred* and *GO MF Pred* graphs contain annotations to GO classes of only two of the three GO sub-domains. Despite being independent of each other, these sub-domains complement each other during protein description. For instance, previous works have found that annotations to the Biological Process and Cellular Component aspects of the functional characterization of interacting proteins can be very informative of their molecular functions (Sousa et al. (2020b)). These graphs are used for model evaluation on its ability to predict for only labels that are classes of the domain for which annotations were removed from the graph, either BP or MF, so model performance for predictions over these graphs are not directly comparable to performance for predictions over the other graphs.

Table 4.4 summarizes the graphs constructed based on their edge selection approaches.

Dataset name	Edge types
"GO full"	GO links + All annotations
"GO train"	GO links + train annotations
"GO BP Pred"	GO slim links + MF & CC annotations
"GO MF Pred"	GO slim links + BP & CC annotations

Table 4.4: Summary of GO graphs built and edge selection approaches.

Node Features

Two different types of node features were used, one for proteins in the PPI graph, another for classes in the GO graph.

1. PPI features

   Original OGB dataset features, obtained from STRING. All edges come with these 8-dimensional features, where each dimension represents the approximate confidence value of a single association type two proteins can have and takes on values between 0 and 1. The larger the value, the more confidence there is on the association. The eight association types pertain to the grouping made by STRING on interactions inferred based on homology, co-expression, co-occurrence, among others (Szklarczyk et al. (2019)).

2. GO features

   The Information Content -based ICseco metric was computed for each GO class, coupled with several other node-level centrality measures that encode information of the GO classes' relative position in the GO hierarchy, as described below:

   (a) Degree: The number of edges connected to one node is defined as its degree. In directed networks these can be further subdivided into incoming degree, outgoing degree and total degree. A node with high degree is well connected and may play a role in maintaining network structure. Thus the number of interactions a node has positively correlates with its importance in the network (Metcalf and Casey (2016)).

   (b) Betweenness: The fraction of the shortest paths between all pairs of nodes that pass through one node or edge, and provides an estimate of the information flow through one node or edge. It is a better indicator for the importance of a gene product than degree centrality (Perez and Germon (2016)).

   (c) Depth (Distance to nearest root): The shortest path possible from a given node to the nearest root of the tree, this can be applied to the GO graph considering its hierarchical nature, similar to a tree's. A root is the topmost class of the top-down hierarchical structure of GO.

   (d) Distance to nearest leaf: The shortest path possible from a given node to the nearest leaf of the graph. A leaf class is the bottom-most class of the top-down hierarchical structure of GO.

4.3.2 Dual-Graph Neural Network

The Dual-Graph Neural Network (DGNN) was proposed to tackle the inclusion of relevant GO information as protein features instead of it being a part of the proteins' neighbourhoods in the PPI network as proposed in the first part of the methodology. This neural network allows the combination of input data from different sources, using different types of neural layers to learn from inputs of different natures, and combining the different extracted features to perform a global prediction based on them.

The DGNN model contains two sub-networks, each of which receives a graph as input. Each input is first processed independently by a graph convolutional branch and then the outputs generated by the convolutional layers of the two branches are merged by concatenating them and processed together by a final fully-connected layer to obtain the global output and a final activation layer (Sigmoid) to flatten the output before giving it to the BCELoss function. The DGNN's sub-networks' parameters are updated separately during backpropagation, unlike Siamese neural networks, where there's weight sharing between the sub-networks. Two DGNNs were tested with the datasets built, one that uses the traditional

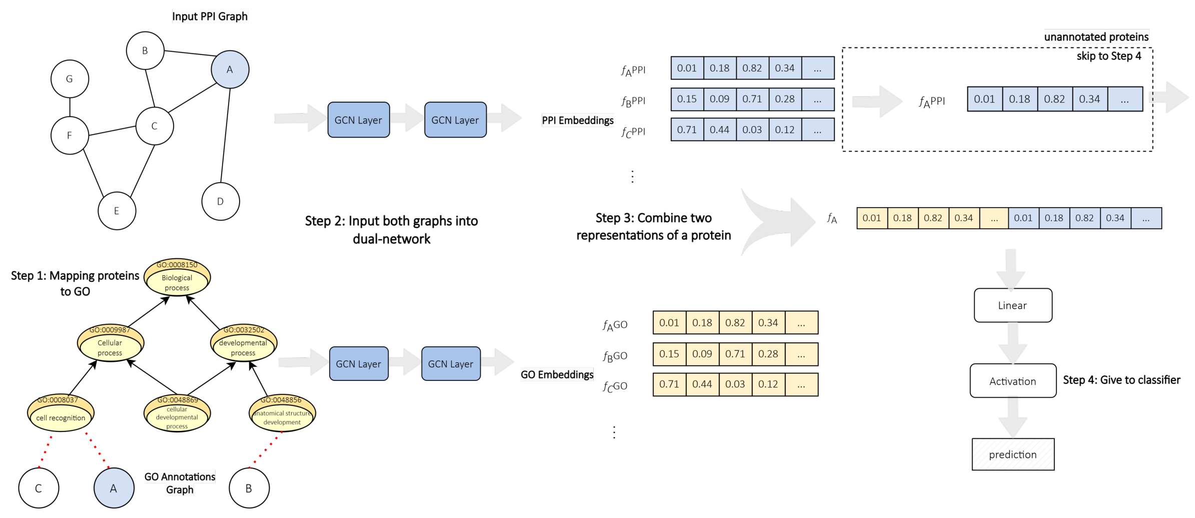

Input PPI Graph
GCN Layer
GCN Layer
PPI Embeddings
unannotated proteins skip to Step 4
Step 2: Input both graphs into dual-network
Step 3: Combine two representations of a protein
Step 1: Mapping proteins to GO
Biological process
Cellular process
developmental process
cell recognition
cellular developmental process
anatomical structure development
GO Annotations Graph
GO Embeddings
Linear
Activation
Step 4: Give to classifier
prediction

Figure 4.2: Dual-input approach methodology.

GCN architecture as sub-networks and another that uses GraphSAGE networks. These architectures were run in Python with DGL on a PyTorch back-end.

The network receives two input graphs, the PPI graph on one side and a GO graph that includes protein annotations to its classes. Each graph is passed through a Graph Convolutional Network -based branch that will determine and output representations for each protein. The branch that convolutes over the PPI network will generate protein representations that take into account both protein node features and their connectivity in the network. The sub-network where the GO knowledge graph is passed through will generate embedding for each protein based on the representation of their annotations to the GO. Each of the resulting protein embeddings from the PPI graph sub-network will then be concatenated to its corresponding embeddings of the GO Graph sub-network. The resulting protein representation will be used by the model to make predictions over protein functionality by assigning one or more of the 112 GO labels to it.

4.3.3 Evaluation

The dual-graph approach was evaluated by combining the DGNN with each of the four graphs produced from exploring the inclusion of different kinds of annotations to the GO (see Table 4.4 of section 4.3.1). These combinations were evaluated on random and species-based protein splitting scenarios as explained in 4.1.1.

The baseline cases were established by testing model performance for classification with a GCN and GraphSAGE singularly over the PPI network. Proteins are classified taking into account only PPI network information as features. These baselines' networks are architecturally identical to the sub-network GCN applied to the PPI graph in the double-input experiment.

Training was done using Adam optimizer and during 1000 epochs with an evaluation step at every 5 epochs.

To evaluate the quality of predictions for the DGNN on each of the proposed experiments and allow for result comparison between the two methodologies of this work the ROC-AUC and Accuracy metrics were utilized once again, with the ROC-AUC remaining the main evaluation metric. The best test scores

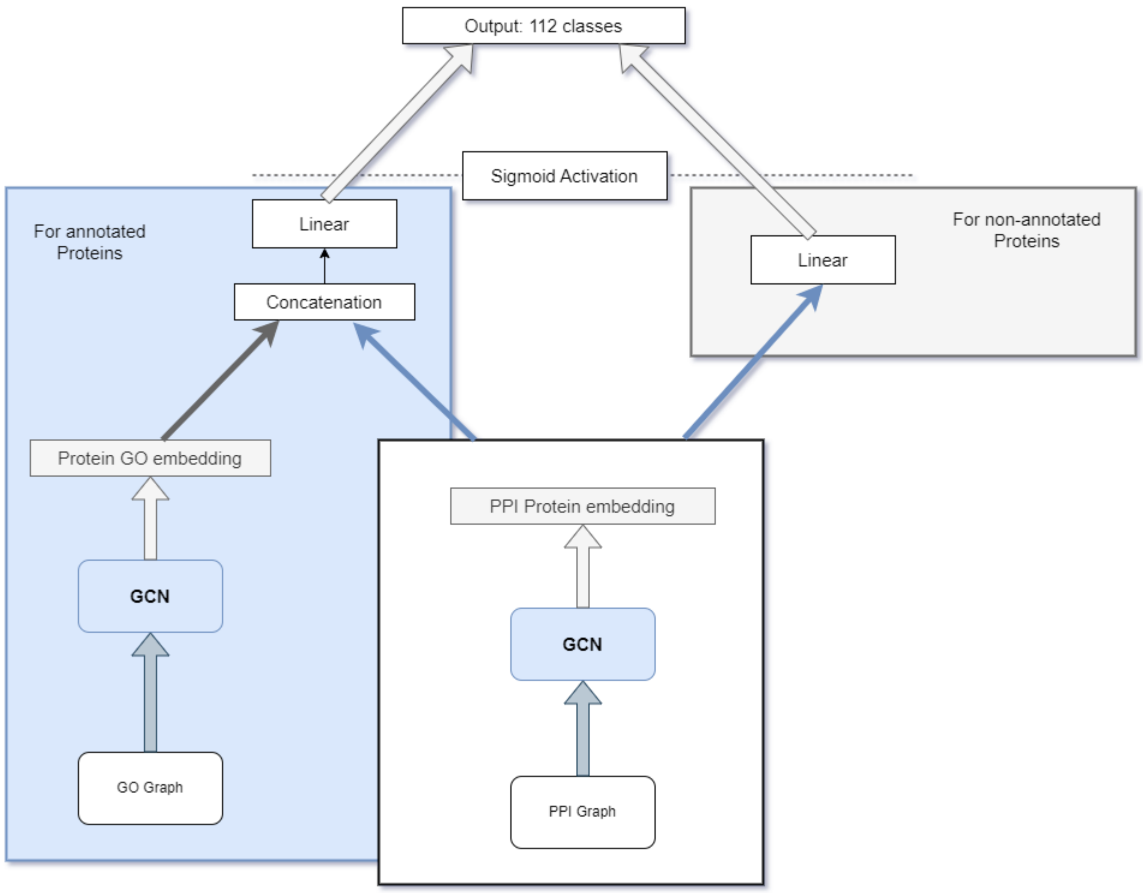


Figure 4.3: Dual-input neural network architecture. The GCN model used in each neural branch can either be the one proposed by Kipf and Welling (2016) or GraphSAGE.

were selected based on the epoch in which the ROC-AUC validation score was the highest.

Chapter 5

Results and Discussion

This section is divided in three parts. The first part presents the results obtained with the methodology implemented for the protein function prediction setting over the PPI-GO graphs. These results are then compared with the baselines. The second part presents the results obtained for the DGNN's prediction over the PPI network's proteins both for the baselines and for the several experiments proposed for this approach, with representations of each protein for both the PPI graph and the GO graph. The partitioning schemes utilized for both methodologies allow for the evaluation of both a cross-species prediction setting, where the model testing steps are made over proteins of a species unknown to the model, and of a global prediction, where the model has prior knowledge of all classification proteins' species. Model evaluation over these two distinct partitioning schemes will address the limitations of predicting functions of proteins whose species are unknown to the model. The third part of this section will see a comprehensive discussion of the results obtained with several of the experiments and evaluation settings carried out in this work.

5.1 Holistic Graph -based Protein Function Prediction

5.1.1 Cross-species Protein function Prediction

Table 5.1 contains the results used for establishing the cross-species prediction evaluation baselines by replicating the OGB team's model evaluation results over the benchmark OGBn-proteins dataset, as published in their website's leaderboard.

Table 5.1: Evaluation results for benchmark PPI network dataset with species-split partitions. Each model evaluation was repeated over 10 runs, results shown are the computed mean value of the set of best results for all runs.

	ROC-AUC	Accuracy
MLP	0.716 (±0.001)	0.964 (±0.000)
MLP+node2vec	0.682 (±0.000)	0.950 (±0.000)
GCN	0.723 (±0.001)	0.927 (±0.000)
GraphSAGE	0.775 (±0.000)	0.901 (±0.000)

The MLP coupled with node2vec embeddings of the PPI network's proteins performed the worst, even when compared to the sole MLP approach, that only leverages PPI -based node features and does not explore the PPI network's structural information. The GraphSAGE neural network produced the highest ROC-AUC results, at 0.775.

Table 5.2 discloses the classification results obtained from evaluating the proposed ML models over two PPI-GO datasets (*PPI-GO full* and *PPI-GO train*), and with the usage of three different node feature representations: a) PPI features (8-dimensional mean values of PPI scores); b) PPI features + GO feature (ICseco score); c) None.

Table 5.2: Classification performance (average and standard deviation of 10 runs) over *PPI-GO full* graph and over *PPI-GO train* graph. ROC-AUC values in bold are an improvement over the baseline in Table 5.1.

	Node Features	PPI-GO full		PPI-GO train	
		ROC-AUC	Accuracy	ROC-AUC	Accuracy
MLP	PPI	0.716 (±0.001)	0.948 (±0.000)	0.716 (±0.004)	0.929 (±0.002)
	PPI + IC_{Seco}	**0.973** (±0.001)	0.970 (±0.001)	0.583 (±0.008)	0.927(±0.009)
MLP	PPI	**0.741** (±0.003)	0.963 (±0,000)	**0.738** (±0.008)	0.930 (±0.003)
+	PPI + IC_{Seco}	**0.970** (±0.001)	0.971 (±0.001)	**0.761** (±0.005)	0.928 (±0.009)
node2vec	None	0.628 (±0.001)	0.951 (±0.001)	0.338 (±0.005)	0.832 (±0.001)
	PPI	**0.779** (±0.002)	0.958 (±0.000)	0.630 (±0.010)	0.918 (±0.002)
GCN	PPI + IC_{Seco}	0.500 (±0.001)	0.939 (±0.001)	0.479 (±0.009)	0.914 (±0.015)
	None	0.381 (±0.001)	0.921 (±0.001)	0.248 (±0.005)	0.850 (±0.001)
	PPI	**0.964** (±0.003)	0.920 (±0.001)	0.721 (±0.012)	0.970 (±0.003)
GraphSAGE	PPI + IC_{Seco}	**0.970** (±0.001)	0.942 (±0.001)	0.660 (±0.017)	0.978 (±0.012)
	None	**0.770** (±0.001)	0.920 (±0.001)	0.432 (±0.008)	0.867 (±0.001)

As seen in Table 5.2, most of the ML models were able to improve their predictive performance over the *PPI-GO full* graph when compared to their evaluation over the benchmark dataset's graph (see Table 5.1), in some cases reaching almost an increase of 0.200 of ROC-AUC metric. This major performance increase proves how the predictor is able to learn functional information from the inclusion of protein annotations to GO classes that are not classification labels, as information regarding already known functions of a given protein can help predict functions it may have that are yet to be discovered. This performance improvement was registered for all three types of protein features used in the experiments. As expected, the MLP model coupled with PPI interaction score features attains similar results to the baselines over most of the PPI-GO datasets, as it is not exploring any GO -related functional information.

Results for the *PPI-GO train* graph showed an overall poor model performance. However, the MLP+Node2Vec approach was able to outperform the baseline results with PPI -based protein features, revealing that the inclusion of known functional aspects in the representations of training proteins allows the MLP predictor to learn meaningful functional patterns to base its predictions on. The removal of PPI interaction score features significantly decreased classification scores for most models.

Table 5.3 displays the results obtained with testing over the *PPI-GO direct* and *PPI-GO slim* graphs. Experiments over these two graphs showed similar predictive behaviour, as prediction results were higher than the baselines' for the majority of model evaluations for at least one of the node feature experiments. However, results were higher for the *PPI-GO direct* graph, hinting that the inclusion of more meaningful annotations to specific GO classes may give more information for the models to base their predictions on when compared to the inclusion of annotations to high-level classes in the *PPI-GO slim* graph. The GCN

Table 5.3: Classification performance (average and standard deviation of 10 runs) over *PPI-GO direct* graph and over *PPI-GO slim* graph. ROC-AUC values in bold are an improvement over the baseline in Table 5.1.

	Node Features	PPI-GO direct		PPI-GO slim	
		ROC-AUC	Accuracy	ROC-AUC	Accuracy
MLP	PPI	0.717 (±0.001)	0.950 (±0.000)	0.717 (±0.003)	0.948 (±0.001)
	PPI + IC_{Seco}	**0.967** (±0.001)	0.979 (±0.000)	**0.962** (±0.004)	0.976 (±0.004)
MLP	PPI	0.644 (±0.025)	0.960 (±0.004)	0.652 (±0.003)	0.955 (±0.006)
+	PPI + IC_{Seco}	**0.970** (±0.001)	0.980 (±0.000)	**0.968** (±0.001)	0.979 (±0.000)
node2vec	None	0.626 (±0.009)	0.973 (±0.001)	0.620 (±0.005)	0.971 (±0.001)
GCN	PPI	0.703 (±0.020)	0.930 (±0.007)	0.712 (±0.011)	0.920 (±0.007)
	PPI + IC_{Seco}	0.429 (±0.004)	0.965 (±0.012)	0.420 (±0.008)	0.963 (±0.003)
	None	0.381 (±0.008)	0.960 (±0.001)	0.432 (±0.005)	0.951 (±0.002)
GraphSAGE	PPI	**0.936** (±0.009)	0.974 (±0.001)	**0.891** (±0.006)	0.969 (±0.004)
	PPI + IC_{Seco}	**0.940** (±0.012)	0.978 (±0.001)	**0.967** (±0.001)	0.979 (±0.000)
	None	0.668 (±0.000)	0.959 (±0.001)	0.633 (±0.005)	0.973 (±0.002)

was the only model that failed to improve on the baseline's results when evaluated over the two graphs. On a side note, the inclusion of an information-content measure as a node centrality attribute (see rows with *PPI + ICseco* node features) obtained the highest results overall.

Table 5.4: Baseline and overall best ROC-AUC prediction scores for each ML algorithm for tasks over *PPI-GO full* graph and over *PPI-GO train* graph and node features used.

		PPI-GO full		PPI-GO train	
	Baseline	ROC-AUC	Node Features	ROC-AUC	Node Features
MLP	0.716 (±0.001)	0.973 (±0.001)	PPI + IC_{Seco}	0.716 (±0,004)	PPI
MLP+node2vec	0.682 (±0.000)	0.970 (±0.001)	PPI + IC_{Seco}	0.761 (±0,005	PPI + IC_{Seco}
GCN	0.723 (±0.001)	0.779 (±0.002)	PPI	0.630 (±0.010)	PPI
GraphSAGE	0.775 (±0.000)	0.970 (±0.001)	PPI + IC_{Seco}	0.721 (±0.012)	PPI

Table 5.4 shows the highest performance results for each of the ML models over the *PPI-GO full* and *PPI-GO train* graphs. The MLP, MLP+node2vec and GraphSAGE -based approaches' ROC-AUC performances improved in 0.200 (20%) or more over the *PPI-GO full* graph with PPI and ICseco scores as protein features when compared to baseline results. The GCN improved predictions, in over 0.050 (5%), over the same graph with PPI scores as features. The MLP+node2vec approach improved its performance in about 0.080 (8%) with predictions over the *PPI-GO train* with PPI and ICseco scores as features.

5.1.2 Global protein function prediction

Table 5.5 contains results for the established global protein function prediction evaluation's baselines, obtained by testing the set of example neural algorithms used in Hu et al. (2020) over the benchmark OGBn-proteins dataset with randomly split protein partitions.

The MLP feed-forward network obtained the lowest ROC-AUC performance score, as it does not

Table 5.5: Evaluation results for benchmark PPI dataset with randomly-split protein partitions.

Algorithm	ROC-AUC	Accuracy
MLP	0.789 (±0.001)	0.895 (±0.000)
MLP+node2vec	0.839 (±0.000)	0.905 (±0.000)
GCN	0.828 (±0.001)	0.902 (±0.001)
GraphSAGE	0.878 (±0.001)	0.914 (±0.001)

utilize any structural information to aid model prediction. The MLP+node2vec obtained a higher performance, potentially hinting at how the inclusion of structural and node connectivity information is useful, in this case for a multi-species setting. The GraphSAGE neural network once again produced the highest ROC-AUC results, at 0.878.

Tables 5.6 and 5.7 display results for a global prediction evaluation scenario where the PPI network's training, validating and testing proteins were split randomly. The model is trained on a multi-species setting and model testing is done over proteins of the same species as in training. Table 5.6 shows the results obtained from evaluating each ML method for protein function prediction over the "PPI-GO full" graph and over the "PPI-GO train" graph. Table 5.7 discloses the results obtained with model evaluation over the "PPI-GO direct" graph and over the "PPI-GO slim" graph.

Table 5.6: Classification performance (average and standard deviation of 10 runs) over the *PPI-GO full* graph and over the *PPI-GO train* graph. ROC-AUC values in bold are an improvement over the baseline in Table 5.5.

	Node Features	PPI-GO full		PPI-GO train	
		ROC-AUC	Accuracy	ROC-AUC	Accuracy
MLP	PPI	0.790 (±0.007)	0.895 (±0.003)	0.790 (±0.010)	0.895 (±0.008)
	PPI + IC_{Seco}	**0.863** (±0.007)	0.955 (±0.003)	0.597 (±0.012)	0.888 (±0.008)
MLP + node2vec	PPI	**0.868** (±0.009)	0.909 (±0.001)	**0.869** (±0.015)	0.909 (±0.007)
	PPI + IC_{Seco}	**0.894** (±0.009)	0.960 (±0.002)	0.807 (±0.015)	0.893 (±0.001)
	None	0.809 (±0.006)	0.968 (±0.002)	0.420 (±0.005)	0.846 (±0.004)
GCN	PPI	**0.854** (±0.010)	0.953 (±0.008)	0.819 (±0.020)	0.939 (±0.010)
	PPI + IC_{Seco}	**0.862** (±0.011)	0.958 (±0.008)	0.817 (±0.013)	0.937 (±0.018)
	None	**0.903** (±0.012)	0.968 (±0.007)	0.496 (±0.018)	0.858 (±0.004)
GraphSAGE	PPI	**0.922** (±0.012)	0.945 (±0.009)	0.762 (±0.018)	0.889 (±0.014)
	PPI + IC_{Seco}	**0.920** (±0.010)	0.949 (±0.008)	0.739 (±0.024)	0.888 (±0.008)
	None	**0.927** (±0.011)	0.923 (±0.001)	0.826 (±0.012)	0.898 (±0.000)

As seen in Table 5.6, in the global prediction evaluation setting most models prediction performances over the *PPI-GO full* graph were able to overachieve the baselines' performances (see 5.5). This demonstrates that the models are leveraging already known functional knowledge of prediction proteins to infer on their unknown functional classes. The GNNs obtained their highest performance when not using any features, uniquely exploring protein connectivity in the graph, while the two MLP algorithms greatly leveraged the inclusion of the PPI and ICseco scores.

Model testing over the *PPI-GO train* graph revealed worse results than those obtained with the *PPI-GO full*, but once again showed that the MLP+node2vec combined approach with PPI score features was able to obtain higher results than those of the equivalent baseline evaluation over the benchmark dataset.

Table 5.7: Classification performance (average and standard deviation of 10 runs) over KG with non-redundant GO annotations and over KG with only protein annotations to the GO Slim generic sub-ontology. Each model evaluation was repeated over 10 runs, results shown are the computed mean value of the set of best results for all runs. ROC-AUC values in bold are an improvement over the baseline in Table 5.5.

	Node Features	PPI-GO direct		PPI-GO slim	
		ROC-AUC	Accuracy	ROC-AUC	Accuracy
MLP	PPI	0.789 (±0.001)	0.895 (±0.000)	0.790 (±0.001)	0.895 (±0.000)
	PPI + IC_{Seco}	**0.821** (±0.001)	0.894 (±0.000)	0.789 (±0.001)	0.896 (±0.000)
MLP + node2vec	PPI	0.837 (±0.000)	0.904 (±0.000)	0.831 (±0.000)	0.904 (±0.000)
	PPI + IC_{Seco}	**0.869** (±0.000)	0.904 (±0.001)	**0.865** (±0.000)	0.906 (±0.000)
	None	0.724 (±0.004)	0.954 (±0.001)	0.732 (±0.004)	0.958 (±0.001)
GCN	PPI	**0.835** (±0.002)	0.900 (±0.006)	**0.830** (±0.003)	0.902 (±0.001)
	PPI + IC_{Seco}	**0.843** (±0.003)	0.905 (±0.001)	**0.831** (±0.007)	0.902 (±0.001)
	None	**0.877** (±0.008)	0.905 (±0.001)	**0.863** (±0.007)	0.905 (±0.001)
GraphSAGE	PPI	**0.945** (±0.001)	0.901 (±0.001)	0.875 (±0.002)	0.899 (±0.003)
	PPI + IC_{Seco}	0.878 (±0.000)	0.896 (±0.000)	0.828 (±0.000)	0.898 (±0.000)
	None	**0.890** (±0.001)	0.905 (±0.000)	0.861 (±0.003)	0.898 (±0.001)

The results obtained for model evaluation on global protein prediction tasks over the two *PPI-GO direct* and *PPI-GO slim* graphs are shown in Table 5.7. Overall, most of the ML models were still able to improve classification performance when compared to the baselines. While the absolute evaluation scores' increase may be smaller than those obtained for predictions with the cross-species partitioning, the global prediction evaluations produced generally higher results, which is to be expected, as the models had previous knowledge of other proteins of the same species as testing proteins'.

Table 5.8 shows the baseline and overall best results for each of the ML approaches over the *PPI-GO full* and *PPI-GO train* graphs for the global protein function prediction -based evaluation setting.

Table 5.8: Baseline and overall best prediction scores for each ML algorithm for tasks over *PPI-GO full* graph and over *PPI-GO train* graph and node features used.

	Baseline	PPI-GO full		PPI-GO train	
		ROC-AUC	Node Features	ROC-AUC	Node Features
MLP	0.789 (±0.001)	0.863 (±0.007)	PPI + IC_{Seco}	0.790 (±0.010)	PPI
MLP+node2vec	0.839 (±0.000)	0.894 (±0.009)	PPI + IC_{Seco}	0.869 (±0.015)	PPI
GCN	0.828 (±0.001)	0.903 (±0.012)	None	0.819 (±0.020)	PPI
GraphSAGE	0.878 (±0.001)	0.927 (±0.011)	None	0.826 (±0.012)	None

The results in Table 5.8 show how model evaluation over the *PPI-GO full* graph for different experiments with node feature inclusion displayed a substantial increase from the baselines' results. The MLP+node2vec was also able to improve prediction performance for the global prediction task over the *PPI-GO train* graph. These results hint that most of the MLP+node2vec also does not require extra functional knowledge for prediction proteins when they are of a species unknown to the model.

5.2 Dual-Graph -based Protein Function Prediction

This section contains the results obtained from testing the two dual-graph neural networks, with either fully GCN-based or GraphSAGE-based sub-networks, over the PPI network and over each of the GO graphs created with different sets of annotations (See Methodology, Table 4.4). The GCN and GraphSAGE DGNN models' evaluations over the benchmark PPI network were used as baselines for result comparison with the proposed dual-graph neural approach. For that, the evaluation produced by OGB over their benchmark dataset with their neural implementations was replicated.

Cross-species Protein function Prediction

Tables 5.9 and 5.10 contain the results obtained for: i) the established baselines; ii) a second baseline case where a final linear layer is applied after the OGB-implemented neural models of the baselines to approximate their architecture to the DGNN network's and account for any effects that an extra linear layer may have on predictive performance; and iii) finally the results pertaining to the evaluation of the DGNN over each of the experiments developed.

Table 5.9's results were obtained using the OGB species-based protein splitting scheme over the training, validation and testing sets, while Table 5.10's results pertain to the evaluation with the same randomly generated protein splitting partitions as those used for the holistic graph approach of the protein prediction problem.

Table 5.9: Evaluation results obtained with dual-graph -based approach for baseline and DGNN -based experiments with species-split partitions. Each evaluation was repeated over 10 runs, results shown are the computed mean value of the set of best results for all runs. * corresponds to the baseline use case where a linear layer is added as a final layer of the OGB's neural implementations. ROC-AUC values in bold are an improvement of the baselines.

DL	Graph	ROC-AUC	Accuracy
GCN	PPI (baseline)	0.728 (±0.001)	0.882 (±0.000)
	PPI (linear*)	0.734 (±0.002)	0.868 (±0.001)
	PPI + GO	**0.951** (±0.006)	0.980 (±0.001)
	PPI + GO train	0.399 (±0.010)	0.978 (±0.001)
	PPI + GO w/o MF annots	0.964 (±0.005)	0.970 (±0.001)
	PPI+ GO w/o BP annots	0.971 (±0.004)	0.964 (±0.000)
GraphSAGE	PPI graph (baseline)	0.772 (±0.001)	0.874 (±0.000)
	PPI graph (linear*)	0.744 (±0.001)	0.921 (±0.000)
	PPI + GO	**0.981** (±0.004)	0.980 (±0.000)
	PPI + GO train	0.681 (±0.007)	0.978 (±0.001)
	PPI + GO w/o MF annots	0.952 (±0.003)	0.953 (±0.001)
	PPI+ GO w/o BP annots	0.979 (±0.003)	0.979 (±0.001)

Comparison of the ROC-AUC results between the *PPI + GO* experiment and the baseline for *PPI graph* (see both "baseline" and the "linear" adaptation in Table 5.9) show that both the GCN and GraphSAGE variants of the dual-graph approach greatly outperformed the baselines in over 0.200 of ROC-AUC. The *PPI + GO train* approach did not produce better results than the baseline's, somehow the inclusion of only training protein annotations in the GO graph seems to bring some noise into the information being learned by the model, as it is not able to properly predict testing proteins' functionality.

Furthermore, although results obtained for *PPI + GO MF pred* and *PPI + GO BP pred* experiments are not directly comparable to the baselines and remaining experiments, it is possible to see how their results were similar to those of prediction for functions pertaining to any of the three GO sub-domains, which proves the model is effectively learning from the inclusion of information of a different biological significance than that which the model will predict.

Global protein function prediction

Table 5.10: Evaluation results obtained with dual-graph -based approach for baseline and DGNN -based experiments over PPI network with random partitions. Each evaluation was repeated over 10 runs, results shown are the computed mean value of the set of best results for all runs. ”*” corresponds to the baseline use case where a linear layer is added as a final layer of the OGB’s neural implementations. ROC-AUC values in bold are an improvement of the baselines.

DL	GO graph	ROC-AUC	Accuracy
	PPI (baseline)	0.828 (±0.002)	0.902 (±0.000)
	PPI (linear*)	0.843 (±0.002)	0.903 (±0.000)
GCN	PPI + GO	**0.878** (±0.006)	0.905 (±0.000)
	PPI + GO train	0.575 (±0.009)	0.888 (±0.001)
	PPI + GO w/o MF annots	0.863 (±0.006)	0.895 (±0.001)
	PPI + GO w/o BP annots	0.891 (±0.006)	0.903 (±0.001)
	PPI (baseline)	0.879 (±0.001)	0.914 (±0.000)
	PPI (linear*)	0.882 (±0.002)	0.915 (±0.000)
GraphSAGE	PPI + GO	**0.917** (±0.003)	0.924 (±0.000)
	PPI + GO train	0.684 (±0.008)	0.888 (±0.001)
	PPI + GO w/o MF annots	0.907 (±0.004)	0.898 (±0.000)
	PPI + GO w/o BP annots	0.911 (±0.003)	0.902 (±0.000)

Through Table 5.10 it can be seen that both the GCN and GraphSAGE variants of the dual-graph approach obtained higher ROC-AUC results with the *PPI + GO* experiment than the baselines tests made over the *PPI graph* (see both ”baseline” and ”linear” adaptation), while the *PPI + GO train* approach once again did not produce better results than the baseline’s.

Model evaluation for the *PPI + GO MF pred* and *PPI + GO BP pred* experiments obtained similar results to those of predicting for functions pertaining to any of the three GO sub-domains. This indicates the model is basing its functional predictions on knowledge obtained from the inclusion of information pertaining to other known functional aspects.

5.3 Discussion

General result comparison showed that for holistic graph and dual-graph -based protein function prediction the overall best performances were obtained with the *PPI-GO full* graph (ROC-AUC of 0.973 for cross-species; ROC-AUC of 0.927 for global prediction) and the *PPI+GO* graph (ROC-AUC of 0.981 for cross-species; ROC-AUC of 0.917 for global prediction), respectively. The two graphs include annotations to both general and more specific classes in the GO hierarchy for both training and testing proteins. As seen for the holistic graph prediction, the predictor greatly benefited from this broad approach of including annotations to the entirety of the GO, even outperforming testing of experiments over the *PPI-GO direct* graph, that excludes non-direct annotations to parent classes of more specific protein GO annotations, and the *PPI-GO slim* graph, that only includes annotations to a set of meaningful general GO classes. However, one of the limitations of these experiments, where the models are trained over graphs that include annotations for testing proteins, is that although in many cases there is some prior knowledge about some functional aspects of a protein while others are still unknown, it can also happen that no function has yet been determined for the proteins predictions are made for.

As such, this work tested for these situations with the experiments made over the *PPI-GO train* and *GO train* graphs, for the holistic (HGNN) and dual-graph (DGNN) prediction problems respectively, that did not include any functional annotations for the prediction proteins; the model only learned functional information for the proteins it is trained on. Result comparison for predictions over these graphs with those of the *PPI-GO full* and *PPI+GO* graphs indicates that while most neural models required the learning of already known functional aspects of a protein to greatly increase their potential to infer its unknown functions, for the holistic graph -based prediction task an approach (MLP coupled with node2vec) was able to train a model from the sole inclusion of other proteins' functional information that was able to improve predictions over its query proteins' functionality. The performance drop for model training with no testing protein annotations is more noticeable in the dual-graph prediction task, likely because the predictor did not have any kind of functional representation in the global learning of the proteins it made predictions for, while in holistic graph prediction, despite not knowing testing proteins' functional information either, it leveraged from including neighboring (interacting) proteins' functional representations in the holistic representation of the testing proteins, that includes mixed PPI and GO-related information.

An even more realistic scenario for predictions over biological knowledge is when some functional aspect of the protein (cellular component it is located in, or biological process it performs in) is already known and the aim is to leverage this information to predict a molecular function (or vice-versa). To study this situation, two sets of experiments were made for the dual-graph prediction approach by predicting molecular function or biological process labels without considering molecular function or biological process annotations respectively. Evaluations for both a cross-species and a global prediction scenario resulted in a very good performance in terms of ROC-AUC, hinting at the approach's ability to learn a model able to leverage other functional sub-domains' information to infer on a given aspect of a protein. This is particularly useful in cases where a protein is poorly characterized for its molecular level functions but there's some knowledge of broader functional information, such as the pathways it is involved in. However, these results are not directly comparable to those of the other experiments carried out in this work, since each model is now learning specific domain-bound labels instead of the classification task's full set of labels.

Result analysis for cross-species protein function prediction evaluation for both holistic and dual-graph prediction problems revealed that the models were able to learn over several model species' proteins and successfully obtain generalizable, species-agnostic knowledge that allowed it to make good predictions for an unknown species' proteins. Usually predictors achieve better results for global prediction evaluation settings, as they are trained to obtain species-specific protein functionality information, while on cross-species prediction not only does the predictor not know the proteins' species but the training over species other than the one for testing may introduce some level of bias that limits the generalization power of the predictor. However, overall comparison of evaluation results for cross-species and global protein prediction showed that, while both prediction scenarios for the proteins of the OGBn-proteins dataset achieved better model performance than the corresponding baselines, the cross-species prediction often achieved better results than the global prediction.

Finally, the comparison of the sets of best results (for the *PPI-GO full* and *PPI+GO* graphs) for the GNN-based approaches of the two methodologies demonstrated that, for the global prediction task, the best performance was obtained with the holistic graph approaches, despite not being much higher than the dual graph's (0.025 of ROC-AUC metric difference for the GCN and 0.010 for GraphSAGE), while for the cross-species prediction task, the dual-graph GNN-based approaches achieved higher results than its counterpart for the GCN, with over 0.200 of ROC-AUC improvement, and 0.011 for GraphSAGE.

The GCN-based DGNN's exceeding performance for the cross-species prediction seemingly indicates it was able to generalize better for unknown species than the holistic graph's. The Dual-Graph approach's predictive power could be due to it being capable of learning a general model from exploring two completely different perspectives of protein functionality, one based on the protein's known functional aspects and another on its interactions with other proteins with known functions, while the holistic graph approach captures a single, full-scale and heterogeneous view of the two perspectives, which may introduce some noise along with the information learned by the model during training. This noise, however, is not particularly noticeable in the global prediction setting, where the holistic graph approach seems better at capturing knowledge of protein similarity within the prediction species.

Chapter 6

Conclusions

Recent biomedical applications of ontologies have led to most of the biological knowledge available being encoded into a structured representation of its meaning and relationships that can be modelled as a graph. With biological knowledge structured in a similar way to that of biological data, e.g. PPI networks, an opportunity to enrich a biological data graph with knowledge about its entities arises. This is where the focus of this work lies; it intends to test if the inclusion of a biological ontology layer into a PPI graph could improve the performance of graph mining approaches that can explore the protein information represented in said ontology.

For that, two different methodologies to including GO-related information were developed and tested using the OGB project's benchmark PPI network dataset. The first methodology saw the construction of several graphs that directly connected the GO graph to the original PPI network through the use of protein GO annotations. This approach was combined with several state-of-the-art ML algorithms for a protein function prediction task. In the second methodology, a multi-input neural architecture that receives both the PPI graph and the GO graph was tested as an alternative approach to incorporating knowledge of protein functionality and combining it with PPI-related information while maintaining them as two separate views of the proteins' characterization instead of combining them in the same graph and potentially creating some noise.

For each of the proposed approaches, several experiments into including using different node features were explored, for both proteins and GO classes. Furthermore, two types of protein distribution schemes were used, to explore ML predictive capabilities when doing a cross-species prediction and global prediction evaluation of the ML models. The evaluation of the two approaches proved they led to an overall increase of model performance when compared to the results obtained with predictions over the OGB team's benchmark, proving how the inclusion of information regarding known protein functions can increase predictive ability over unknown functions. Result comparison between the two partitioning schemes used for model evaluation also showed that the ML approaches successfully predicted for a protein species they had no prior knowledge of, which in the biological context is very important, as there are few model organisms for which protein functionality is well-known and several poorly-known species for which knowledge of protein functionality is scarce.

To conclude, this work produced two different methodologies into combining GO information with PPI data. The first methodology sees the combining of ML algorithms with approaches of including the

GO with PPI networks. The second methodology encompasses the creation of a novel Graph Neural Network approach that combines representations of PPI and the GO to learn a general model for protein function that explores background knowledge. The OGB team's benchmark was used for evaluating and comparing the existing ML approaches.

The multi-label classification task raised class imbalance issues, with classes for which practically all proteins were annotated and classes that severely lacked protein annotations, and did not allow the exploring of label dependency, as the task was treated as a set of binary classifiers that were trained independently for each GO class. These do not account for hierarchical relations between GO classes in a protein function prediction problem.

As such, taking into account these limitations it would be interesting to use datasets with between-class balancing and/or explore ML variants that best suit a protein function prediction problem, for example by implementing a Hierarchical Multi-label classification task instead, that allows for multi-label prediction while accounting for label dependency. Future endeavors may also include expanding the type of PPI network -based prediction tasks in which similar approaches to including the GO into PPI data will be employed (e.g. link prediction tasks).

Acronyms

CNN	Convolutional Neural Networks
DAG	Directed Acyclic Graph
DGNN	Dual-Graph Neural Network
DL	Deep Learning
GCN	Graph Convolutional Network
GNN	Graph Neural Network
GO	Gene Ontology
KG	Knowledge Graph
ML	Machine Learning
NN	Neural Network
PPI	Protein-Protein Interaction

Bibliography

Abdi, H. (1994). A neural network primer. *Journal of Biological Systems*, 02:247–281.

Ahmed, A., Shervashidze, N., Narayanamurthy, S. M., Josifovski, V., and Smola, A. (2013). Distributed large-scale natural graph factorization. *Proceedings of the 22nd international conference on World Wide Web*.

Albawi, S., Mohammed, T. A., and Al-Zawi, S. (2017). Understanding of a convolutional neural network. *2017 International Conference on Engineering and Technology (ICET)*, pages 1–6.

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. (2000). Gene ontology: tool for the unification of biology. *Nature genetics*, 25(1):25–29.

Barabási, A.-L., Gulbahce, N., and Loscalzo, J. (2011). Network medicine: a network-based approach to human disease. *Nature reviews genetics*, 12(1):56–68.

Bard, J. B. L. and Rhee, S. Y. (2004). Ontologies in biology: design, applications and future challenges. *Nature Reviews Genetics*, 5:213–222.

Barros, C. D. T., da Silva, D. N. R., and Porto, F. (2021). Machine learning on graph-structured data. *Anais Estendidos do XXXVI Simpósio Brasileiro de Banco de Dados (SBBD Estendido 2021)*.

Belkin, M. and Niyogi, P. (2001). Laplacian eigenmaps and spectral techniques for embedding and clustering. In *NIPS*.

Bellomarini, L., Sallinger, E., and Vahdati, S. (2020). Knowledge graphs: the layered perspective. In *Knowledge Graphs and Big Data Processing*, pages 20–34. Springer, Cham.

Bonetta, R. and Valentino, G. (2020). Machine learning techniques for protein function prediction. *Proteins: Structure, Function, and Bioinformatics*, 88(3):397–413.

Cao, S. (2016). deep neural network for learning graph representations.

Cao, S., Lu, W., and Xu, Q. (2015). Grarep. pages 891–900.

Chami, I., Abu-El-Haija, S., Perozzi, B., Ré, C., and Murphy, K. P. (2020). Machine learning on graphs: A model and comprehensive taxonomy. *ArXiv*, abs/2005.03675.

Clark, W. T. and Radivojac, P. (2011). Analysis of protein function and its prediction from amino acid sequence. *Proteins: Structure, Function, and Bioinformatics*, 79(7):2086–2096.

Dayhoff, J. E. (1989). Neural network architectures: An introduction.

Dimou, A. (2020). Creation of knowledge graphs. In *Knowledge Graphs and Big Data Processing*, pages 59–72. Springer.

Doosti, B., Naha, S., Mirbagheri, M., and Crandall, D. J. (2020). Hope-net: A graph-based model for hand-object pose estimation. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6607–6616.

Egaña, M., Bechhofer, S., Lord, P., Sattler, U., and Stevens, R. (2007). Understanding and using the meaning of statements in a bio-ontology: Recasting the gene ontology in owl. *BMC bioinformatics*, 8:57.

Fa, R., Cozzetto, D., Wan, C., and Jones, D. T. (2018). Predicting human protein function with multi-task deep neural networks. *PLoS ONE*, 13.

Fahlman, S. (1999). An empirical study of learning speed in back-propagation networks.

Geleta, D., Nikolov, A., Edwards, G., Gogleva, A., Jackson, R., Jansson, E., Lamov, A., Nilsson, S., Pettersson, M., Poroshin, V., Rozemberczki, B., Scrivener, T., Ughetto, M., and Papa, E. (2021). Biological insights knowledge graph: an integrated knowledge graph to support drug development. *bioRxiv*.

Ghamrawi, N. and McCallum, A. (2005). Collective multi-label classification. In *International Conference on Information and Knowledge Management*.

Gilmer, J., Schoenholz, S. S., Riley, P. F., Vinyals, O., and Dahl, G. E. (2017). Neural message passing for quantum chemistry. In Precup, D. and Teh, Y. W., editors, *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 1263–1272. PMLR.

Grover, A. and Leskovec, J. (2016). node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 855–864.

Guarino, N. (1997). Understanding, building and using ontologies. *Int. J. Hum.-Comput. Stud.*, 46:293–310.

Gui, T., Dong, X., Li, R., Li, Y., and Wang, Z. (2015). Identification of hepatocellular carcinoma–related genes with a machine learning and network analysis. *Journal of Computational Biology*, 22(1):63–71.

Hamilton, W., Ying, Z., and Leskovec, J. (2017). Inductive representation learning on large graphs. *Advances in neural information processing systems*, 30.

Hamilton, W. L. (2020). Graph representation learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*.

Han, J. (2002). How can data mining help bio-data analysis? In *BIOKDD*.

Han, J. and Moraga, C. (1995). The influence of the sigmoid function parameters on the speed of back-propagation learning. In *International Work-Conference on Artificial and Natural Neural Networks*.

Hanif, M. S. and Aono, M. (2010). Metric of intrinsic information content for measuring semantic similarity in an ontology. In *APCCM*.

Hertz, J. A., Krogh, A., and Palmer, R. G. (1991). Introduction to the theory of neural computation. In *The advanced book program*.

Hu, W., Fey, M., Zitnik, M., Dong, Y., Ren, H., Liu, B., Catasta, M., and Leskovec, J. (2020). Open graph benchmark: Datasets for machine learning on graphs. *Advances in neural information processing systems*, 33:22118–22133.

Hu, Y. H., Hwang, J. N., and Hwang, J.-N. (2005). Handbook of neural network signal processing. *IEEE Transactions on Neural Networks*, 16:780–780.

Ito, T., Chiba, T., Ozawa, R., Yoshida, M., Hattori, M., and Sakaki, Y. (2001). A comprehensive two-hybrid analysis to explore the yeast protein interactome. *Proceedings of the National Academy of Sciences*, 98(8):4569–4574.

Janev, V., Graux, D., Jabeen, H., and Sallinger, E. (2020). *Knowledge graphs and big data processing*. Springer Nature.

Jaramillo-Garzón, J., Castellanos-Dominguez, G., and Perera, A. (2016). Applicability of semi-supervised learning assumptions for gene ontology terms prediction. *Revista Facultad de Ingeniería, Universidad de Antioquia*, 2016:19–32.

Joos, T. O. and Bachmann, J. (2005). The promise of biomarkers: research and applications. *Drug discovery today*, 10(9):615–616.

King, R. D., Karwath, A., Clare, A., and Dehaspe, L. (2000). Accurate prediction of protein functional class from sequence in the mycobacterium tuberculosis and escherichia coli genomes using data mining. *Yeast*, 17(4):283–293.

Kingma, D. P. and Ba, J. (2015). Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980.

Kipf, T. N. and Welling, M. (2016). Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*.

Kiryakov, A., Popov, B., Ognyanoff, D., Manov, D., Kirilov, A., and Goranov, M. (2003). Semantic annotation, indexing, and retrieval. In *International Workshop on the Semantic Web*.

Kong, K., Li, G., Ding, M., Wu, Z., Zhu, C., Ghanem, B., Taylor, G., and Goldstein, T. (2020). Flag: Adversarial data augmentation for graph neural networks. *ArXiv*, abs/2010.09891.

Kulmanov, M., Khan, M. A., and Hoehndorf, R. (2018). Deepgo: predicting protein functions from sequence and interactions using a deep ontology-aware classifier. *Bioinformatics*, 34(4):660–668.

Kumar, S. K. (2017). On weight initialization in deep neural networks. *CoRR*, abs/1704.08863.

Latouche, P. and Rossi, F. (2015). Graphs in machine learning. an introduction. *ArXiv*, abs/1506.06962.

LeCun, Y., Bengio, Y., and Hinton, G. (2015). Deep learning. *Nature*, 521:436–444.

Li, L., Doroslovaki, M., and Loew, M. H. (2020a). Approximating the gradient of cross-entropy loss function. *IEEE Access*, 8:111626–111635.

Li, X., Hu, Y., Sun, Y., Hu, J., Zhang, J., and Qu, M. (2020b). A deep graph structured clustering network. *IEEE Access*, 8:161727–161738.

Lin, N., Wu, B., Jansen, R., Gerstein, M., and Zhao, H. (2004a). Information assessment on predicting protein-protein interactions. *BMC bioinformatics*, 5(1):1–11.

Lin, N., Wu, B., Jansen, R., Gerstein, M. B., and Zhao, H. (2004b). Information assessment on predicting protein-protein interactions. *BMC Bioinformatics*, 5:154 – 154.

Lv, Q., Ma, W., Liu, H., Li, J., Wang, H., Lu, F., Zhao, C., and Shi, T. (2015). Genome-wide protein-protein interactions and protein function exploration in cyanobacteria. *Scientific reports*, 5(1):1–12.

MacLean, F. (2021). Knowledge graphs and their applications in drug discovery. *Expert Opinion on Drug Discovery*, 16:1057 – 1069.

Martin, S., Szekely, B., and Allemang, D. (2021). *The Rise of the Knowledge Graph*. O'Reilly Media, Incorporated.

Martins, I. C., Gomes-Neto, F., Faustino, A. F., Carvalho, F. A., Carneiro, F. A., Bozza, P. T., Mohana-Borges, R., Castanho, M., Almeida, F. C. L., Santos, N. C., and da Poian, A. T. (2012). The disordered n-terminal region of dengue virus capsid protein contains a lipid-droplet-binding motif. *The Biochemical journal*, 444 3:405–15.

Metcalf, L. and Casey, W. (2016). *Cybersecurity and applied mathematics*. Syngress.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., and Dean, J. (2013). Distributed representations of words and phrases and their compositionality. *ArXiv*, abs/1310.4546.

Minsky, M. and Papert, S. (1969). An introduction to computational geometry. *Cambridge tiass., HIT*, 479:480.

Mitchell, T., Carbonell, J., and Michalski, R. (1986). Machine learning. In *The Springer International Series in Engineering and Computer Science*, volume 12.

Nariai, N., Kolaczyk, E. D., and Kasif, S. (2007). Probabilistic protein function prediction from heterogeneous genome-wide data. *Plos one*, 2(3):e337.

Niepert, M., Ahmed, M., and Kutzkov, K. (2016). Learning convolutional neural networks for graphs. *ArXiv*, abs/1605.05273.

Parthasarathy, S., Tatikonda, S., and Ucar, D. (2010). A survey of graph mining techniques for biological datasets. In *Managing and Mining Graph Data*.

Patil, A. and Nakamura, H. (2005). Filtering high-throughput protein-protein interaction data using a combination of genomic features. *BMC bioinformatics*, 6(1):1–13.

Perez, C. and Germon, R. (2016). Graph creation and analysis for linking actors: Application to social data. In *Automating open source intelligence*, pages 103–129. Elsevier.

Perozzi, B., Al-Rfou, R., and Skiena, S. (2014). Deepwalk: online learning of social representations. *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*.

Read, J. and Perez-Cruz, F. (2015). Deep learning for multi-label classification.

Rehman, S. U., Khan, A. U., and Fong, S. (2012). Graph mining: A survey of graph mining techniques. In *Seventh International Conference on Digital Information Management (ICDIM 2012)*, pages 88–92. IEEE.

Rosenblatt, F. (1958). The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, 65(6):386.

Ruby, U. and Yendapalli, V. (2020). Binary cross entropy with deep learning technique for image classification. *International Journal of Advanced Trends in Computer Science and Engineering*, 9.

Rumelhart, D. E., Hinton, G. E., and Williams, R. J. (1986). Learning representations by back-propagating errors. *Nature*, 323:533–536.

Ryan, D. P. and Matthews, J. M. (2005). Protein–protein interactions in human disease. *Current opinion in structural biology*, 15(4):441–446.

Sarkar, D. and Saha, S. (2019). Machine-learning techniques for the prediction of protein–protein interactions. *Journal of biosciences*, 44(4):1–12.

Schmidhuber, J. (2015). Deep learning in neural networks: An overview. *Neural networks : the official journal of the International Neural Network Society*, 61:85–117.

Sousa, R. T., Silva, S., and Pesquita, C. (2020a). Evolving knowledge graph similarity for supervised learning in complex biomedical domains. *BMC bioinformatics*, 21(1):1–19.

Sousa, R. T., Silva, S., and Pesquita, C. (2020b). Evolving knowledge graph similarity for supervised learning in complex biomedical domains. *BMC Bioinformatics*, 21.

Sureyya Rifaioglu, A., Doğan, T., Jesus Martin, M., Cetin-Atalay, R., and Atalay, V. (2019). Deepred: automated protein function prediction with multi-task feed-forward deep neural networks. *Scientific reports*, 9(1):1–16.

Szklarczyk, D., Gable, A. L., Lyon, D., Junge, A., Wyder, S., Huerta-Cepas, J., Simonovic, M., Doncheva, N. T., Morris, J. H., Bork, P., et al. (2019). String v11: protein–protein association networks with increased coverage, supporting functional discovery in genome-wide experimental datasets. *Nucleic acids research*, 47(D1):D607–D613.

Uschold, M. and Grüninger, M. (1996). Ontologies: principles, methods and applications. *The Knowledge Engineering Review*, 11:93 – 136.

Velickovic, P., Cucurull, G., Casanova, A., Romero, A., Lio', P., and Bengio, Y. (2018). Graph attention networks. *ArXiv*, abs/1710.10903.

Vidal, M.-E., Endris, K. M., Jozashoori, S., Karim, F., and Palma, G. (2019). Semantic data integration of big biomedical data for supporting personalised medicine. In *Current Trends in Semantic Web Technologies: Theory and Practice*, pages 25–56. Springer.

Wu, Z., Pan, S., Chen, F., Long, G., Zhang, C., and Philip, S. Y. (2020). A comprehensive survey on graph neural networks. *IEEE transactions on neural networks and learning systems*, 32(1):4–24.

Xia, F., Sun, K., Yu, S., Aziz, A., Wan, L., Pan, S., and Liu, H. (2021). Graph learning: A survey. *IEEE Transactions on Artificial Intelligence*, 2:109–127.

Xu, K., Hu, W., Leskovec, J., and Jegelka, S. (2018). How powerful are graph neural networks? *arXiv preprint arXiv:1810.00826*.

Yon Rhee, S., Wood, V., Dolinski, K., and Draghici, S. (2008). Use and misuse of the gene ontology annotations. *Nature Reviews Genetics*, 9(7):509–515.

Zhang, J., Shi, X., Xie, J., Ma, H., King, I., and Yeung, D.-Y. (2018). Gaan: Gated attention networks for learning on large and spatiotemporal graphs. *ArXiv*, abs/1803.07294.

Zhang, L., Song, H., Aletras, N., and Lu, H. (2022). Node-feature convolution for graph convolutional networks. *Pattern Recognition*, 128:108661.

Zhang, S., Chen, H., Liu, K., and Sun, Z. (2009). Inferring protein function by domain context similarities in protein-protein interaction networks. *BMC bioinformatics*, 10(1):1–6.

Zhou, Z.-H. (2021). *Machine learning*. Springer Nature.

Zupan, J. (2017). Artificial neural network. In *ACM SIGSPATIAL International Workshop on Advances in Geographic Information Systems*.