PREDICTING DISEASE-GENE ASSOCIATIONS VIA MACHINE LEARNING

MAKİNE ÖĞRENMESİ İLE GEN-HASTALIK İLİŞKİSİ TAHMİNİ

OSMAN ONUR KUZUCU

PROF. DR. TUNCA DOĞAN
Supervisor

Submitted to
Graduate School of Science and Engineering of Hacettepe University
as a Partial Fulfillment to the Requirements
for the Award of the Degree of Master of Science
in Computer Engineering

June 2024

ABSTRACT

PREDICTING DISEASE-GENE ASSOCIATIONS VIA MACHINE LEARNING

Osman Onur KUZUCU
Master of Science, Computer Engineering
Supervisor: Prof. Dr. Tunca DOĞAN
June 2024, 132 pages

In the quest to elucidate disease etiology and develop advanced diagnostic and treatment tools, knowing disease-gene relationships is of great importance. Traditional approaches based on manual curation fall short due to limited scalability and precision. On the other hand, graph neural networks (GNN) enable the analysis of complex relational data within biological networks. Although the GNN-based methods developed to date have produced positive results in predicting unknown biological relationships, there is a current need to develop new models with high prediction performance and generalisation capabilities for usability in biology and medicine. In this thesis study, we propose GLADIGATOR (Graph Learning bAsed DIsease Gene AssociaTiOn pRediction), a deep learning model designed with the encoder-decoder architecture to predict disease-gene associations. GLADIGATOR creates a heterogeneous graph that primarily integrates two types of biological components, genes and diseases, and the connections between them. The model was trained using gene-gene, disease-disease and gene-disease relationships existing in source biological databases, as well as protein sequence representations generated by the Prot-T5[1] protein language model and disease representations generated by the BioBert[2] language model, as node feature vectors. As the outcome of the analyses conducted, it was observed that

GLADIGATOR had superior prediction accuracy. Additionally, the model was positioned as the highest performer among 14 different disease-gene association prediction methods. Literature-driven studies on selected predictions have confirmed the biological relevance of predicted novel associations and highlighted the effectiveness of the GNN-based approach in identifying potential candidate genes for specific diseases. These results may provide valuable information for discovering new drugs as a result of future experimental validation analyses. GLADIGATOR has not only enriched computational approaches developed for disease-gene association prediction but also emphasised the transformative abilities of GNNs in biomedical research by potentially accelerating the discovery of new biological relationships.

Keywords: disease gene association prediction, biological graph, machine learning, graph neural networks

ÖZET

MAKİNE ÖĞRENMESİ İLE GEN-HASTALIK İLİŞKİSİ TAHMİNİ

Osman Onur KUZUCU
Yüksek Lisans, Bilgisayar Mühendisliği
Danışman: Prof. Dr. Tunca DOĞAN
Haziran 2024, 132 sayfa

Hastalık etiyolojisini aydınlatma ve ileri tanı ve tedavi araçları geliştirme arayışında, hastalık-gen ilişkilerinin bilinmesi büyük önem taşımaktadır. Manuel kürasyona dayalı geleneksel yaklaşımlar, sınırlı ölçeklenebilirlik ve hassasiyet nedeniyle yetersiz kalmaktadır. Öte yandan, çizge sinir ağları (ÇSA), biyolojik ağlar içindeki karmaşık ilişkisel verinin analizini mümkün kılmaktadır. Bugüne kadar geliştirilmiş olan ÇSA tabanlı yöntemler bilinmeyen biyolojik ilişkilerin tahmini konusunda olumlu sonuçlar üretmiş olsa da, biyoloji ve tıp alanlarında kullanılabilirlik için tahmin performansı genelleme yetenekleri yüksek seviyede olan yeni modellerin geliştirilmesine ihtiyaç duyulmaktadır. Bu çalışmada, hastalık-gen ilişkilerini tahmin etmek için kodlayıcı-kod çözücü mimarisi ile tasarlanmış bir derin öğrenme modeli olan GLADIGATOR'u (Çizge Öğrenme Tabanlı Hastalık Gen İlişkilendirmesi) öneriyoruz. GLADIGATOR, öncelikle genler ve hastalıklar olarak iki biyolojik bileşen tipini ve aralarındaki bağlantıları entegre eden heterojen bir çizge oluşturur. Model, kaynak biyolojik veri tabanlarında mevcut gen-gen, hastalık-hastalık ve hastalık-gen ilişkilerinin yanı sıra Prot-T5[1] protein dil modeli tarafından oluşturulan protein dizi temsillerini ve BioBert[2] dil modeli tarafından oluşturulan hastalık temsillerini düğüm özellik vektörleri olarak kullanılarak eğitilmiştir. Yapılan analizlerde, GLADIGATOR'un üstün tahmin doğruluğuna sahip olduğu gözlenmiştir. Ayrıca, model, 14 farklı hastalık-gen ilişki tahmini yöntemiyle karşılaştırıldığında en yüksek performansı sergileyen model olarak konumlanmıştır. Seçili gen-hastalık tahmin örnekleri üzerinde yapılan literatür odaklı çalışmalar, tahmin edilen yeni ilişkilerinin biyolojik uygunluğunu doğrulamıştır ve belirli hastalıklar için potansiyel aday genlerin belirlenmesinde ÇSA temelli yaklaşımın etkinliğinin altını çizmiştir. Bu sonuçlar, ileride gerçekleştirilecek deneysel doğrulama analizleri sonucunda, yeni ilaçların keşfedilmesi için değerli bilgiler sağlayabilir. GLADIGATOR yalnızca hastalık-gen ilişkisi tahmini için geliştirilen hesaplamalı yaklaşımları zenginleştirmekle kalmamış, aynı zamanda da yeni biyolojik ilişkilerinin ortaya çıkarılmasını potansiyel olarak hızlandırarak biyomedikal araştırmalarda ÇSA'nın dönüştürücü yeteneğini vurgulamıştır.

Anahtar Kelimeler: hastalık gen ilişki tahmini, biolojik çizgeler, makine öğrenmesi, çizge sinir ağları

CONTENTS

TABLES

FIGURES

ABBREVIATIONS

GLADIGATOR	:	**G**raph **L**earning-b**A**sed **DI**sease **G**ene **A**ssocia**TiO**n p**R**ediction
CTD	:	**C**omparative **T**oxicogenomics **D**atabase
CGI	:	**C**ancer **G**enome **I**nterpreter
MGD	:	**M**ouse **G**enome **D**atabase
RGD	:	**R**ad **G**enome **D**atabase
HPO	:	**H**uman **P**henotype **O**ntology
LHGDN	:	**L**iterature-derived **H**uman **G**ene **D**isease **N**etwork

1. INTRODUCTION

The modern healthcare industry is witnessing a paradigm shift, driven by the synergy of science and technology. This shift is evident in the evolution of healthcare practices, marked by the introduction of innovative devices and treatment methods [16]. As a result, continuous innovation in the healthcare industry becomes a crucial factor for sustaining life [16].

A disease is defined as a condition that interferes with the body's normal physiological functions [17]. Diseases can be of various types and are often the result of multiple factors. Infections, genetic predisposition, environmental factors, dietary habits, lifestyle factors, and immune system issues can contribute to disease development [17]. The treatment methods, such as medication, surgery, physical therapy, dietary and lifestyle changes, depend on the type, severity, and cause of the disease [18]. Strategies like vaccination, healthy diet, regular exercise, hygiene practices, and avoiding risky behaviors are employed to prevent disease development or minimize risk factors [18].

Diseases caused by one or more mutations in an individual's genetic material are referred to as diseases of genetic origin [19]. These diseases are often linked to genetic traits that are inherited from parents to children [19]. Genetic diseases can present in various ways, and their symptoms, severity, and progression can differ significantly from one disease to another [20]. These mutations can impact or alter the production of proteins that affect the body's normal function. For example, in cystic fibrosis, mutations in the CFTR gene disrupt the body's normal mucus production and can cause serious problems in the respiratory and digestive systems [21]. Genetic testing plays a crucial role in identifying individuals who are susceptible to certain genetic diseases and can also aid in diagnosing, treating, and preventing diseases [22]. Research is ongoing to better understand genetic diseases and develop treatments [23].

An understanding of the genetic mechanisms of this disease provides valuable insights into the pathophysiology of this disease. This allows for the development of potential treatment strategies and improvement of existing treatments [23]. For example, targeted treatments can be developed based on a specific genetic mutation, or the diseased gene can be repaired or replaced using gene editing techniques [23].

Identifying the genetic causes of genetically based diseases also plays an important role in determining family disease risk and personalizing health management strategies for the individual [22]. This helps patients and their families understand their risks and protect their health by taking preventative measures if necessary [22].

Understanding the underlying genetic causes of diseases of genetic origin contributes to scientific research and medical knowledge [23]. This sets the direction for future research and treatment developments, guiding efforts to prevent and treat future diseases. Therefore, identifying the causes of genetic diseases is an important step to improve the quality of life of patients and improve public health [23].

Machine learning, a part of artificial intelligence, enables computers to learn from data. It employs algorithms to find and forecast patterns from a large amount of data [24]. These algorithms learn by studying features and relationships in a dataset and try to predict future data or make decisions based on the information gained from these studies [24]. This sub-field of the artificial intelligence is applied in classification, regression, clustering, dimensionality reduction, bounded space, and many other areas [24]. For instance, a classification problem seeks to assign a data sample to a specific class, while a regression problem seeks to predict a continuous output based on the data. Clustering seeks to separate data points into groups based on similarities [24].

Machine learning models are applied in a variety of areas, such as healthcare, finance, automotive, retail, security, and more [24]. For instance, the algorithms are successfully used in disease diagnosis and treatment, stock price prediction, autopilot systems for car drivers, customer segmentation, and personal assistants etc. [24].

In the realm of data science and artificial intelligence, algorithms are undergoing rapid evolution [24]. These evolving algorithms are constantly improved and optimized to solve more complex problems and make more accurate predictions. As a result, this area is having a huge impact on many aspects of modern technology and is expected to become even more popular in the future [24].

In the realm of healthcare, researchers are developing models to predict disease-gene associations and enhance disease diagnosis. Leveraging large datasets, this field excels at uncovering patterns and relationships, making it a potent tool for disease prediction. [24]. Learning models can gather information from various data sources to predict diseases. Various sources, such as databases, patient records, test results, imaging scans, and genetic data, can be used to train these models and improve their accuracy. Analyzing this data can identify specific symptoms or disease risk factors, which may be essential for early diagnosis and treatment.

Prediction of disease-gene association is important for drug design, penalization healthcare etc. This healthcare transformation can improve the quality of life of patients and increase the efficiency of the healthcare system by providing new opportunities for better disease management and healthcare [25].

1.1. Scope Of The Thesis

This thesis delves into the utilization of graph convolutional networks (GCNs) for predicting disease-gene associations. The research encompasses several key areas:

The initial phase of the research involves *Data Collection and Preprocessing*. Heterogeneous biological data, encompassing gene sequences, disease phenotypes, and established disease-gene associations, were gathered from public repositories and databases that everybody can access. It is aimed for researchers to conduct research on these datasets and easily compare the methodologies they have developed. Data preprocessing techniques are employed to maintain data quality and uniformity.

The subsequent phase of the research is centered on the *Graph Construction*. The amassed biological data were utilized to create a heterogeneous graph representation. Graph data structures were intended to be used in this research because they contain relational information. In this graph, nodes symbolize genes and diseases, while edges represent the relationships between them. Various graph construction strategies to encapsulate different types of biological interactions and associations were explored in this research.

The core objective of this thesis is *Model Development*. The aim is to design and implement a novel GCN-based framework for predicting disease-gene links. The model's architecture incorporates an encoder-decoder structure, facilitating the extraction of informative latent representations from the heterogeneous biological graph data. The research will pay special attention to optimizing the model parameters and selecting suitable hyperparameters to boost predictive performance.

The next phase is *Evaluation and Validation*. The performance of the proposed GCN models were thoroughly evaluated using standard metrics for binary classification tasks, such as the area under the receiver operating characteristic curve (AUC-ROC) and precision-recall curve analysis. Furthermore, the biological relevance of the predicted disease-gene associations will be validated through a literature review and comparison with existing biomedical knowledge.

The thesis also includes *Case Studies and Applications*, demonstrating the practical utility of the proposed GCN models in predicting novel disease-gene associations. The aim of this part, the identified candidate genes were analyzed to shed light on their potential roles in disease pathogenesis and therapeutic targeting.

Finally, the thesis concludes with a discussion on *Future Directions and Implications*. This discussion highlights potential future research directions and implications arising from the proposed GCN-based approach. Areas for further investigation include the exploration of advanced graph neural network architectures, integration of multi-omics data modalities, and extension to other biomedical applications such as drug repurposing and precision medicine. The discussion underscores the implications of the research findings for advancing our understanding of disease biology and accelerating translational research efforts.

In summary, this thesis seeks to further the development of computational methods for predicting disease-gene associations and to offer valuable insights into the intricate relationship between genes and diseases in the context of precision medicine and biomedical research.

2. CONTRIBUTIONS

This thesis introduces several advancements in the field of disease-gene association prediction using machine learning. The first advancement is the introduction of a *Feature Selection*. This innovative approach is created to select relevant features from genomic data, enhancing the accuracy of disease-gene association prediction models.

The second contribution of this thesis is the exploration of *Deep Learning Architectures and their Encode-Decode Operations* for disease-gene association prediction. It delves into the application of various deep learning architectures, including graph neural networks (GNNs) and graph convolutional networks (GCNs), to genomic data. These architectures are renowned for their ability to capture complex patterns and are used to extract significant insights from gene expression data, thereby enhancing the predictive performance of the models.

The third contribution of this thesis is the *Refinement of Evaluation Metrics*. AUC-ROC and Pr-Auc values constitute the performance parameters in relationship prediction models. In this study, popular evaluation measurements such as accuracy and F1 score were used. These measurements have been carefully adapted to the context of disease-gene prediction. These adapted metrics provide a more nuanced assessment of model performance specific to this domain.

The fourth contribution of this thesis is the *Application to Real-World Data*. The proposed approach is applied to the dataset. The predicted results of the relationships in this data set and in the section reserved for testing are compared with the real results. Comparison results highlight the effectiveness of the method in identifying potential disease-gene associations.

In conclusion, this thesis enhances the understanding of disease-gene associations and provides practical insights for researchers and clinicians in the field of personalized medicine.

2.1. Thesis Structure

This thesis is structured as follows:

- Chapter 1 elucidates the motivation, contributions, and the scope of the thesis.
- Chapter 2 offers an overview of the fundamental concepts related to genes, diseases, and machine learning.
- Chapter 3 presents a review of the existing literature on disease-gene association prediction.
- Chapter 4 details the methodologies employed in this research.
- Chapter 5 showcases the results derived from this research.
- Chapter 6 provides a discussion on the outcomes of the proposed methods.
- Chapter 7 concludes the thesis and suggests potential avenues for future research.

3. BACKGROUND OVERVIEW

3.1. Data

3.1.1. Genes

Genes, often the carriers of an organism's hereditary traits, are molecular entities [26]. In organisms, genetic information is stored in chromosomes. DNAs are found in these chromosomes. Genes are embedded in DNA molecules. They regulate numerous biological processes throughout the life of the organism. Figure 3.1 summarizes the flow of genetic information from chromosomes to genes.[27].

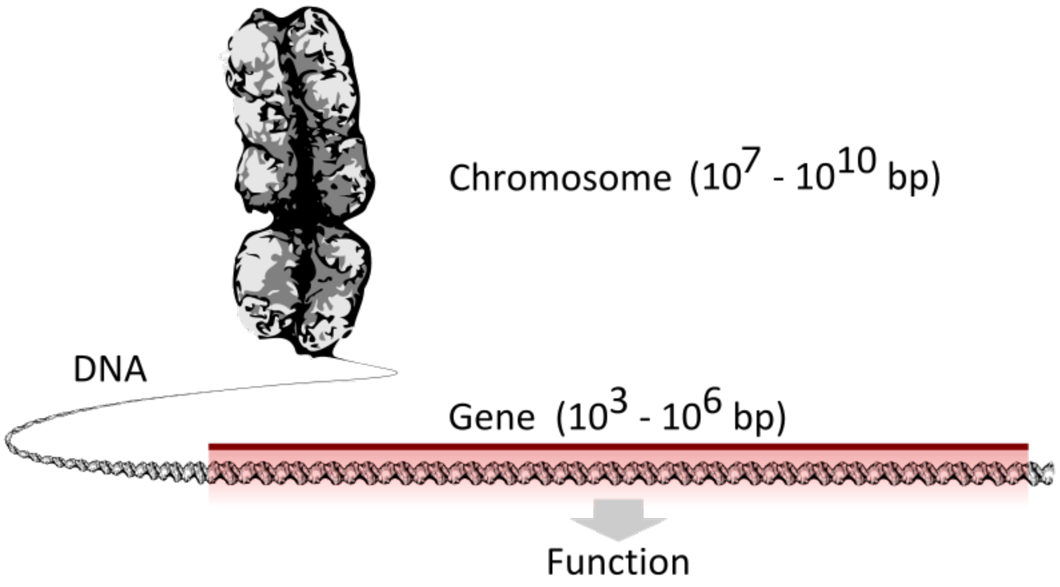


Figure 3.1 Chromosome to Gene Sequence[4]

The primary functions of genes encompass:

- Coding for protein synthesis within the cells of the body. This involves the conveyance of genetic information from DNA to RNA, known as transcription, and subsequently from RNA to protein, referred to as translation [26].

- Regulating the activity of other genes within the body, thereby controlling specific cellular processes and overseeing crucial functions such as development, growth, and the organism's response to environmental conditions [27].

- Determining the inherited traits of the organism. The parents' genetic material defines the child's genotype and phenotype and is transmitted to the next generation [26].

- Contributing to genetic diversity and evolutionary changes among species. Processes like mutation, genetic recombination, and natural selection result in alterations in the gene pool, leading to species evolution over time [27].

The structure and function of genes are central to genetic research, and genetics has significant applications in various fields, including medicine, agriculture, environment, and biotechnology.

A genetic sequence is a particular order of nucleotides present in an organism's DNA or occasionally RNA. DNA is composed of four distinct nucleotides: **adenine** (A), **thymine** (T), **guanine** (G), and **cytosine** (C), as shown in Figure 3.2. In RNA, **uracil** (U) replaces thymine [28].

Each gene, a DNA sequence, encodes a specific characteristic or function of an organism. Genetic sequences hold genetic data that significantly influence an organism's growth, development, and functionality [28].

Genetic sequences are pivotal in genetic research and molecular biology. They aid in identifying particular genes, analyzing gene expression, diagnosing genetic diseases, and determining species relationships. Technological advancements have made gene sequencing techniques faster and more sensitive, enabling a more thorough and detailed examination of gene sequences. This progress has broadened the scope of genetic sequencing applications in numerous fields, including medicine, agriculture, environment, and biotechnology [29].

The association between genes and proteins is a fundamental concept in molecular biology. It refers to the process by which genetic information is used to produce proteins in an

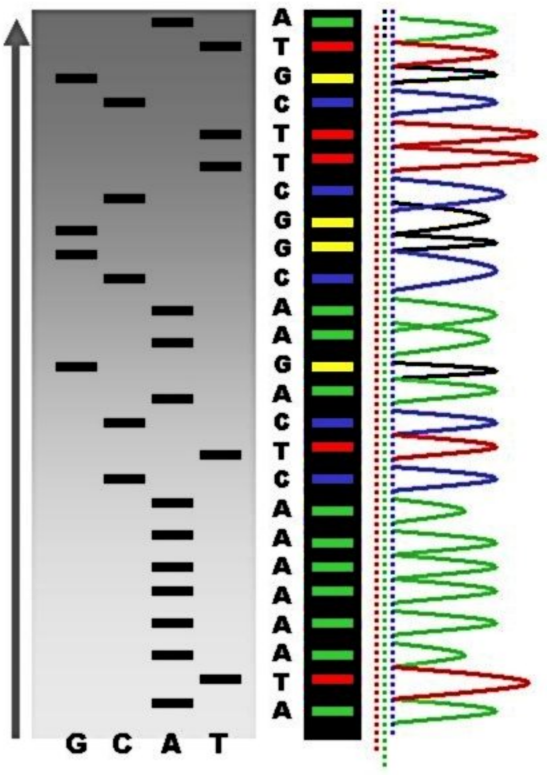


Figure 3.2 DNA Sequencing Sample [5]

organism's cells. Genes, which are specific segments of DNA, carry the instructions for the cellular functions of an organism. Proteins, on the other hand, are responsible for the structure of the body and act as catalysts for various biochemical reactions [30].

The process by which genes contribute to protein synthesis is known as the central dogma of molecular biology. This process involves three main steps. In the first step, the DNA sequence of a gene is transcribed into an RNA molecule, known as messenger RNA (mRNA), by an enzyme called RNA polymerase.

In the second step, the transcribed mRNA molecule undergoes further modifications to become a mature mRNA molecule. This involves processes such as the splicing of the start and stop sequences of the mRNA (kappa) and the removal of non-coding sequences known as introns (splicing).

In the final step, the mature mRNA molecule interacts with cellular structures known as ribosomes. The ribosomes recognize specific sequences, known as codons, in the mRNA molecule. These codons dictate the exact sequence of amino acids that the ribosome synthesizes. During this process, the amino acids are linked together to form polypeptide chains, which eventually fold into functional proteins. This process is illustrated in Figure 3.3.

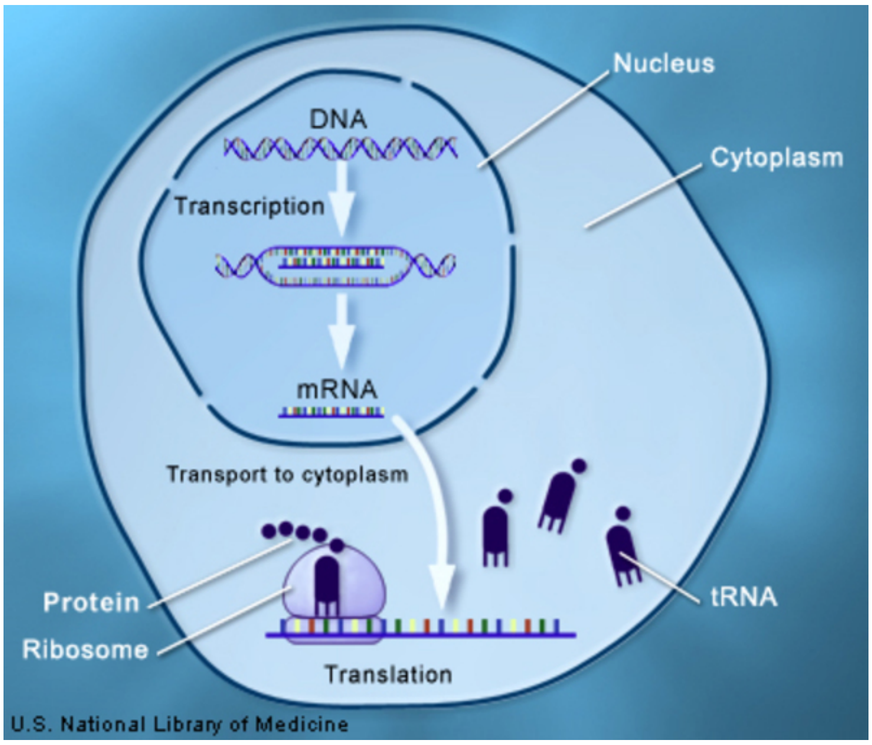


Figure 3.3 Protein Synthesis Mechanism [6]

Through this process, the genetic information encoded in DNA is translated into proteins via mRNA. These proteins can serve various roles in the body, from providing structural support to acting as catalysts for biochemical reactions. The association between genes and proteins is a key mechanism that determines the biological functions and phenotypes of organisms [31].

3.1.2. Diseases

Disease is a term that encompasses a variety of conditions that interfere with the normal functioning of the body. The causes, types, and contributing factors of diseases are diverse and numerous [17].

The origins of disease are multifaceted, encompassing genetic factors, various types of infections (such as bacterial, viral, fungal, or parasitic), environmental influences (like contaminated air, water, or food), lifestyle choices (including diet, smoking, alcohol consumption, and level of physical activity), hormonal imbalances, and issues with the immune system [32]. An example of a disease-causing bacteria, specifically the one responsible for tuberculosis, is depicted in Figure 3.4.

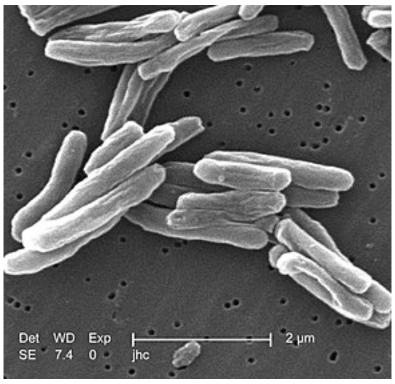


Figure 3.4 Microscopic View of Tuberculosis Bacteria [7]

The Figure 3.4 shows a microscopic view of tuberculosis bacteria. These bacteria are rod-shaped and are the cause of the disease tuberculosis. The “2 µm” scale indicates the size of the bacteria, which is typical for this type of microorganism.

Diseases can be categorized in a multitude of ways. Some examples include infectious diseases like influenza and tuberculosis, genetic diseases such as cystic fibrosis and thalassemia, cancer, metabolic diseases like diabetes, neurological diseases including Alzheimer’s and Parkinson’s, cardiovascular diseases, and mental health disorders like depression and anxiety [33] [34] [35].

The symptoms exhibited by a disease are largely dependent on the specific type and severity of the disease. These may range from pain, fever, and cough, to nausea, vomiting, diarrhea, headache, fatigue, shortness of breath, muscle weakness, memory loss, rashes, and itching.

The treatment approach for a disease is determined by its type, severity, and cause. This could involve medications, surgery, physical therapy, changes in diet and lifestyle, psychotherapy, radiation therapy, or chemotherapy [36].

Many diseases can be prevented, or at least their risk factors can be mitigated, through various prevention strategies. These include vaccination, regular exercise, maintaining a healthy diet, practicing good hygiene, regular health check-ups, and avoiding risky behaviors [37].

The management and treatment of diseases are typically overseen by a healthcare professional and are tailored to the individual patient's condition and needs [38].

3.1.3. Disease-Gene Relation

Genes are known to significantly influence the onset of numerous diseases. Studies in human genetics have shed light on the role of genes in the etiology, symptomatology, and treatment of diseases [39].

Certain diseases are directly linked to genetic factors and can be inherited from parents to offspring due to specific alterations in an individual's genetic composition. Examples of such genetic diseases include cystic fibrosis, thalassemia, and Huntington's disease [40, 41]. A map of diseases on different human chromosomes is depicted in Figure 3.5. In addition, A single sample disease is given for each chromosome in Figure 3.5.

The susceptibility to certain diseases is associated with an individual's genetic predisposition. This implies that specific gene variants or combinations thereof can elevate the risk of disease. For instance, complex diseases like certain cancers or heart diseases may involve the interplay of multiple genes [42].

Genetic factors can also affect an individual's response to medications. The field of pharmacogenetics investigates the impact of specific drugs on their efficacy and safety based on an individual's genetic traits. This knowledge can aid in personalizing medication dosages and minimizing side effects [43].

Genetic testing serves as a tool for identifying genetic diseases or predispositions. These tests can be utilized to evaluate disease risk, facilitate early diagnosis, and guide treatment strategies [44]. Gene therapy, a promising approach in the treatment of genetic diseases,

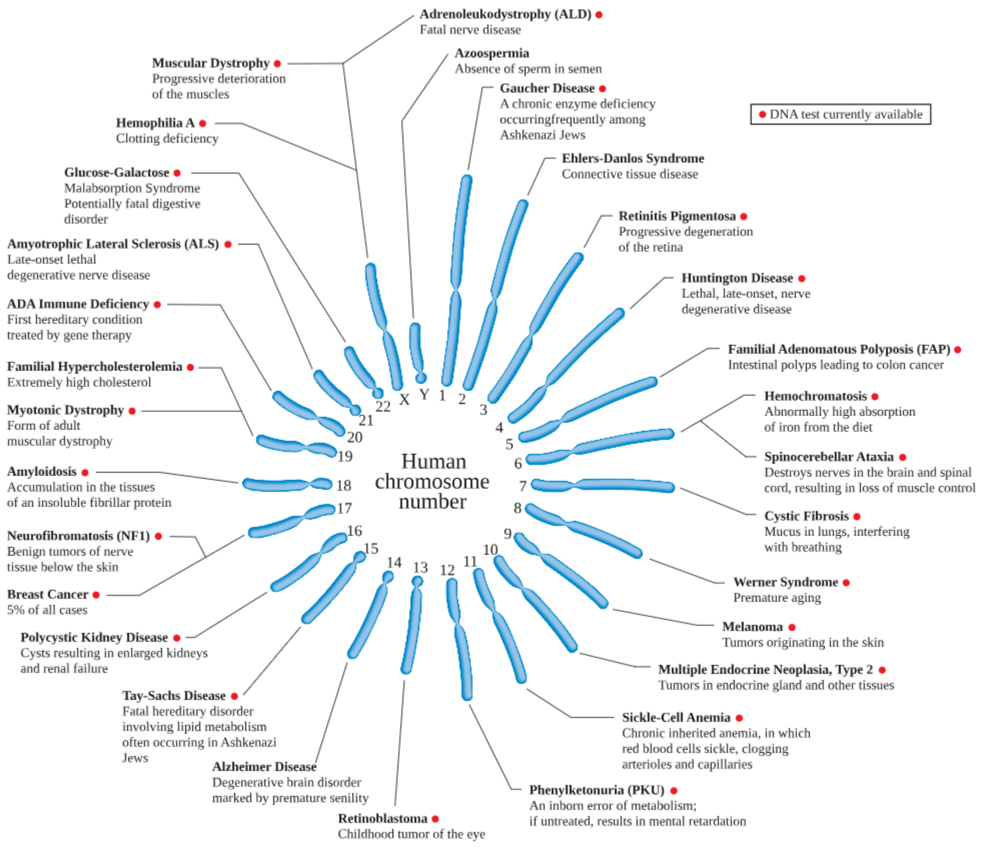

Adrenoleukodystrophy (ALD) ●
Fatal nerve disease
Azoospermia
Absence of sperm in semen
Gaucher Disease ●
A chronic enzyme deficiency occurringfrequently among Ashkenazi Jews
● DNA test currently available
Muscular Dystrophy ●
Progressive deterioration of the muscles
Hemophilia A ●
Clotting deficiency
Glucose-Galactose ●
Malabsorption Syndrome
Potentially fatal digestive disorder
Ehlers-Danlos Syndrome
Connective tissue disease
Retinitis Pigmentosa ●
Progressive degeneration of the retina
Amyotrophic Lateral Sclerosis (ALS) ●
Late-onset lethal degenerative nerve disease
Huntington Disease ●
Lethal, late-onset, nerve degenerative disease
ADA Immune Deficiency ●
First hereditary condition treated by gene therapy
Familial Adenomatous Polyposis (FAP) ●
Intestinal polyps leading to colon cancer
Familial Hypercholesterolemia ●
Extremely high cholesterol
Hemochromatosis ●
Abnormally high absorption of iron from the diet
Myotonic Dystrophy ●
Form of adult muscular dystrophy
Spinocerebellar Ataxia ●
Destroys nerves in the brain and spinal cord, resulting in loss of muscle control
Amyloidosis ●
Accumulation in the tissues of an insoluble fibrillar protein
Human chromosome number
22 X Y 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21
Cystic Fibrosis ●
Mucus in lungs, interfering with breathing
Neurofibromatosis (NF1) ●
Benign tumors of nerve tissue below the skin
Breast Cancer ●
5% of all cases
Werner Syndrome ●
Premature aging
Polycystic Kidney Disease ●
Cysts resulting in enlarged kidneys and renal failure
Melanoma ●
Tumors originating in the skin
Multiple Endocrine Neoplasia, Type 2 ●
Tumors in endocrine gland and other tissues
Tay-Sachs Disease ●
Fatal hereditary disorder involving lipid metabolism often occurring in Ashkenazi Jews
Sickle-Cell Anemia ●
Chronic inherited anemia, in which red blood cells sickle, clogging arterioles and capillaries
Alzheimer Disease
Degenerative brain disorder marked by premature senility
Phenylketonuria (PKU) ●
An inborn error of metabolism; if untreated, results in mental retardation
Retinoblastoma ●
Childhood tumor of the eye

Figure 3.5 A Sample Hereditary Disease on Each Human Chromosomes [8]

seeks to address genetic disorders by substituting or repairing defective genes [45]. Given these considerations, it is evident that genetic information significantly influences our understanding, diagnosis, and treatment of diseases [46].

A deeper comprehension of the genetic underpinnings of diseases paves the way for the development of personalized medical approaches and enhances the effectiveness of disease management.

3.1.4. Disease-Gene Association Prediction

Predicting disease-gene associations is a critical endeavor in bioinformatics and computational biology. It involves the identification of potential genetic factors that may contribute to the susceptibility or resistance of individuals to certain diseases. The goal is to enhance our understanding of the genetic underpinnings of diseases and to facilitate the development of personalized medicine approaches.

Mathematically, the problem can be formulated as follows. Let G be the set of genes and D be the set of diseases. We are interested in finding a function $f : G \times D \to \mathbb{R}$ that predicts the strength of association between a gene $g \in G$ and a disease $d \in D$. The function f is typically learned from a dataset of known disease-gene associations using machine learning techniques.

The prediction function f can take various forms, depending on the approach used. For instance, in a probabilistic framework, f might represent the probability that gene g is associated with disease d, which can be expressed as $P(g \text{ is associated with } d)$. Alternatively, in a regression framework, f could represent a continuous score reflecting the strength of the association.

To build the prediction model, various types of data can be utilized, including genetic sequence data, disease description, disease name and known disease-gene associations from biomedical literature. The integration of such heterogeneous data sources is often achieved through the construction of a disease-gene network, where nodes represent genes and diseases, and edges represent known associations.

The performance of the prediction model is evaluated using metrics such as accuracy, precision, recall, and the area under the receiver operating characteristic curve (AUC-ROC). These metrics help in assessing the model's ability to correctly predict associations and in comparing different predictive models.

3.1.5. Graph & Graph Theory

A *graph* is a mathematical term that refers to a structural arrangement of different connections or relationships, and it is also a data structure. A graph comprises points or objects, which are referred to as "nodes", and "edges" or connections that link these nodes [47]. An example of a graph is depicted in Figure 3.6.

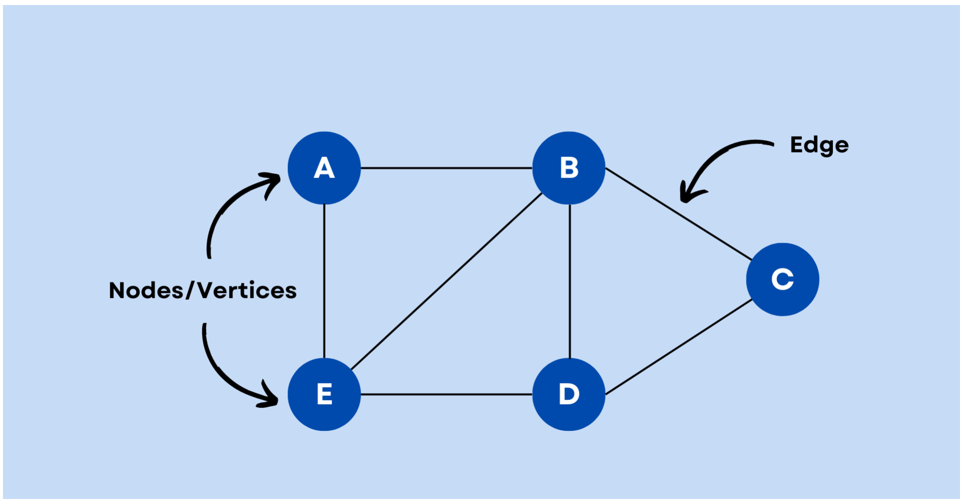


Figure 3.6 Undirected Graph Sample [9]

Graph contains two main components that are nodes and edges. The basic units of a graph is node. Nodes can represent anything, and they are typically labeled with some kind of identifier. Edges are connections between nodes. Edges can be directed or undirected, and they can be weighted or unweighted. Nodes and edges are shown in detail in Figure 3.6.

Graphs can be represented in a variety of ways and have numerous applications. They are used in the analysis of social networks, communication networks, road networks, computer networks, gene editing, and database structures [48].

The fundamental properties of a graph include nodes and edges. Nodes are the basic points or objects of the graph. They can represent users on a social network or cities on a road network. Edges, on the other hand, are the connections that link nodes together. They represent associations or interactions between nodes. For instance, edges can represent friendship links in a social network or paths in a road network.

A graph can either be directed or undirected. This depends on whether the edges follow a certain direction from one node to another. In a directed graph, edges have a direction from one point to another. However, in an undirected graph, this direction does not exist. Some types of graphs can indicate the strength or importance of connections by assigning ”weights” or values to edges. For example, the length of trips in a road network or the strength of friendship ties in a social network can be expressed as weights.

Graph theory, a branch of mathematics, is utilized to model and analyze objects and their relationships in a mathematical manner. It concentrates on structures that are formed by points, referred to as nodes or vertices, and lines that connect these points, known as edges [47].

Graph theory finds its applications in a multitude of fields. One such field is network analysis, where it is used in social networks, communication networks, computer networks, and transmission networks. In this context, nodes typically symbolize entities such as individuals or computers, while edges symbolize the relationships or connections between these entities.

Another significant application of graph theory is in the realm of algorithms and optimization. It forms the foundation of numerous algorithms and aids in solving many problems such as the shortest path problem, flow network problem, and nearest neighbor search problem.

Graph theory also forms the basis of many data structures. For instance, trees, which are a special type of structure in graph theory, are commonly used in computer science. Furthermore, graph theory can be employed to solve sorting problems. Various techniques based on graph theory are used in sorting algorithms and sorting problems.

Graph theory is a crucial tool in many different application areas and provides the capability to model and solve many complex problems [48]. Hence, it finds its applications in various fields such as computer science, engineering, social sciences, business, and others.

3.2. Machine Learning

Machine learning (ML), a subset of artificial intelligence, is concerned with the creation and study of statistical algorithms that can learn from data and generalize to unseen data, thereby performing tasks without explicit instructions. Machine learning is an extension of artificial intelligence. In recent times, generative artificial neural networks have surpassed many previous methods in terms of performance. Machine learning models encompass various learning approaches such as supervised learning, unsupervised learning, and reinforcement learning. These approaches differ from each other based on the feedback mechanism in the training process [49].

3.2.1. Neural Networks & Machine Learning

The structure of machine learning models is not fixed and depends on the algorithm employed. However, a general outline of the structure of a standard supervised learning model, which is one of the most prevalent categories of machine learning models, is as follows:

The model's input data is taken from the **Input Layer**. In this layer, each feature or input variable in the dataset is associated with a node. For instance, in images, each node can represent one pixel; in the case of text data, each node can represent a word or a character.

The **Hidden Layers**, situated between the input and output layers, are responsible for identifying relationships and patterns in the data. Deep learning models, having multiple hidden layers, can learn extremely intricate patterns.

The **Output Layer** generates the model's predictions or outputs using the patterns it identifies in the input data. The structure of the output layer is determined by the type of problem being solved. For instance, in binary classification tasks, a single node may reflect the probability of falling into a single class, whereas in multi-class classification tasks, multiple nodes may reflect the probability of falling into each class.

Each node applies an activation function to the weighted sum of its inputs in the hidden layers and sometimes in the output layer. The model gains nonlinearity from activation functions, enabling it to capture complex patterns. Tanh, sigmoid and ReLU are examples of common activation functions [50].

Sigmoid activation function solves vanishing gradient problem for shallow networks. Primarily usage of the sigmoid activation funcion is in output layers for binary classification problems. Sigmoid activation function's output range between 0 to 1. Sigmoid activation calculation formula is mentioned in Equation 1. Sigmoid activation function plotting is mentioned in Figure 3.7.

$$\sigma(z) = \frac{1}{1+e^{-z}} \tag{1}$$

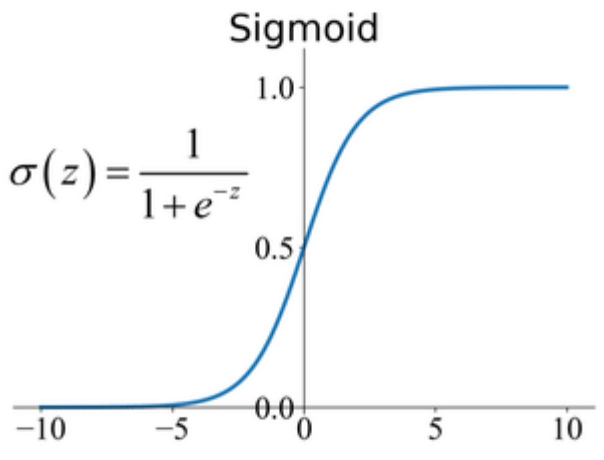


Figure 3.7 Sigmoid Plot [10]

Tanh activation function is similar to sigmoid but with zero-centered output. Tanh activation function often preferred over sigmoid in hidden layers due to zero-centered output. Also, can be used in output layers for certain regression or binary classification problems. Tanh activation function output range is between -1 to 1. Tanh activation calculation formula is mentioned in Equation 2. Tanh activation function plotting is mentioned in Figure 3.8.

$$tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} = \frac{1-e^{-2x}}{1+e^{-2x}} \tag{2}$$

Relu activation function is simple and computationally efficient. Relu activation function is popular activation function for hidden layers in deep neural networks due to efficiency and avoiding vanishing gradients. ReLu activation calculation formula is mentioned in Equation 3. ReLu activation function plotting is mentioned in Figure 3.9.

$$Relu(z) = max(0, z) \tag{3}$$

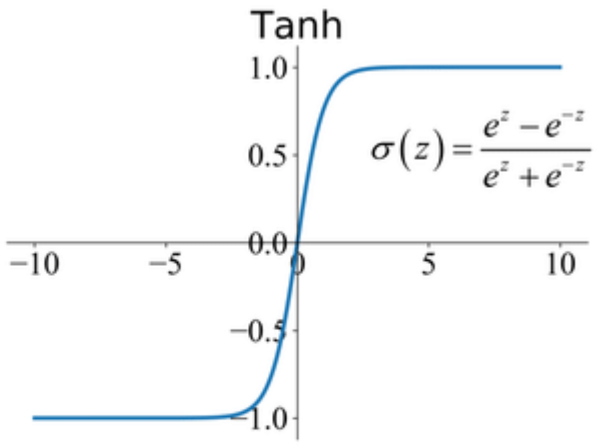


Figure 3.8 Tanh Plot [10]

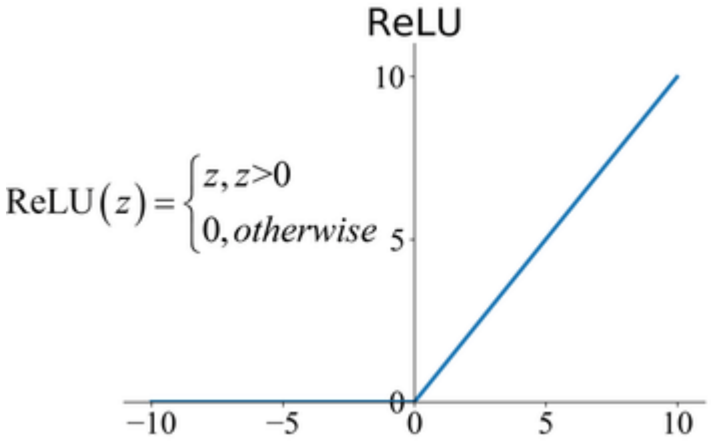


Figure 3.9 ReLu Plot [10]

The learning parameters of the model, **Biases and Weights**, are discovered during training. Biases allow the model to account for biases or shifts in the data, while weights indicate the strength of connections between nodes in neighboring layers. The model adjusts these parameters during training to reduce the discrepancy between targets and its predictions [49].

The **Loss Function** calculates how well the model's predictions align with the actual target values. Minimizing this loss function is the training target for the model. For regression tasks, the mean square error is a common loss function; for classification tasks, the cross-entropy loss is used.

The **Optimization Algorithm** is used to update the weights and biases of the model during training to minimize the loss function. Commonly used optimization methods include gradient descent and its variations, including Adam, AdamW, RAdam, NAdam, and stochastic gradient descent (SGD).

Optimization algorithms play a crucial role in the realms of deep learning and machine learning [51]. A brief overview of these algorithms is as follows:

Stochastic Gradient Descent (SGD) is a fundamental optimization algorithm used during model training. It calculates the gradient (or gradient approximation) for each training example and takes a step against the gradient. This updates the parameters of the model towards the minimized loss function. However, direct SGD may encounter problems such as fluctuation and slow progress in the training process [51].

Adaptive Moment Estimation (Adam) is an enhanced version of SGD, known for its speed and efficiency. Adam, an optimization algorithm, keeps track of the first moment (mean) and second moment (standard deviation) of the gradient. These moments enable the steps to be adapted by following the behavior of the gradients at previous times [51].

AdamW is an advanced version of the Adam optimization algorithm that adds a regularization term by multiplying the weights. This can assist in updating weights more consistently and training a model that is more resistant to overfitting [51].

Rectified Adam (RAdam) is another modification of the Adam optimization algorithm. RAdam uses a type of adaptive multiplication regularization to smooth out gradients at the start of training. This can lead to a more balanced step at the start of training and faster convergence [51].

Nesterov-accelerated Adaptive Moment Estimation (NAdam) is a combined version of Nesterov Momentum and the Adam algorithm. Nesterov Momentum allows direction of momentum using the estimated position of the gradient at a future time step. NAdam aims to achieve faster and more stable convergence by combining these features with the Adam algorithm [51].

These optimization algorithms are extensively used in the training and optimization of deep learning models. Each algorithm has its strengths and weaknesses, and the selection of the one that provides the best performance for a specific problem or dataset is often determined through trial and error.

Regularization Techniques such as L1 and L2 regularization, dropout, and batch normalization are frequently used to prevent overfitting, which occurs when the model learns to memorize training data rather than generalizing to unknown data.

The structure of a machine learning model can vary significantly depending on variables such as the type of data used, the difficulty of the task, and the specific algorithm used. An example of the machine learning model structure is mentioned in Figure 3.10.

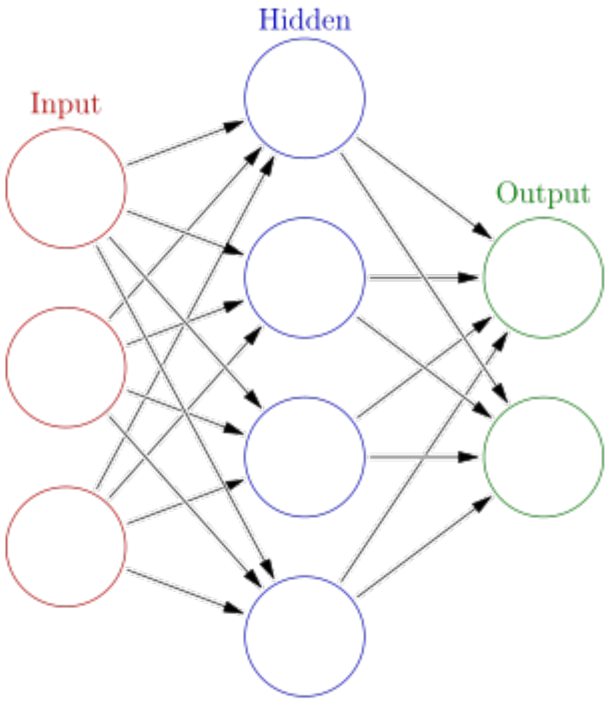


Figure 3.10 Sample Neural Network Architecture[11]

3.2.2. Convolutional Neural Networks

Convolutional Neural Networks, commonly known as CNNs, are a class of deep neural networks highly effective for processing data that has a grid-like topology, such as images.

Their architecture is inspired by the organization of the animal visual cortex and is designed to automatically and adaptively learn spatial hierarchies of features from visual data.[52] Convolutional neural network structure sample is mentioned in Figure 3.11.

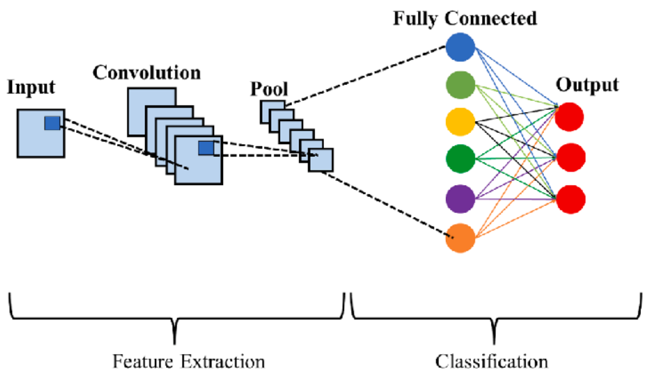


Figure 3.11 Sample Convolutional Neural Network Architecture [12]

At the heart of a CNN is the convolutional layer, where multiple filters are applied to the input to create feature maps. These filters are capable of detecting edges, colors, textures, and other features. The convolutional operation captures the local dependencies within the input, and through the depth of the network, more complex patterns are recognized [53].

Following the convolutional layers, activation functions like ReLU introduce non-linearities, enabling the network to learn complex patterns. Pooling layers are then used to reduce the dimensionality of the feature maps, which decreases the computational load and improves the network's robustness to variations in the input [53].

The high-level features extracted by the convolutional and pooling layers are then flattened and fed into fully connected layers, which act as a classifier. The output layer typically uses a softmax function to provide a probability distribution over the target classes [54].

CNNs learn through a process called backpropagation. During training, the network makes predictions, compares them to the actual labels, calculates the error, and adjusts the weights of the filters to minimize this error. This process is repeated over many iterations, allowing the network to improve its predictions [54].

The strength of CNNs lies in their ability to learn features directly from the data without the need for manual feature extraction, making them particularly suited for image recognition tasks. They have been successfully applied to various applications, including image and video recognition, image classification, medical image analysis, and many others, revolutionizing the field of computer vision.

3.2.3. Graph Convolutional Network Structure

The Graph Convolutional Network (GCN) is a type of neural network designed to work directly with graph-structured data. Graphs are digital structures consisting of nodes (representing elements) and edges (representing connections or relationships between elements). GCNs have recently gained popularity due to their applicability in various tasks, including graph-related tasks such as node classification, link prediction, and graph classification [55].

In a GCN, data is represented as a graph, typically an adjacency matrix (A) and a node feature matrix (X). The adjacency matrix (A) represents the relationships between the nodes in the graph, where each element (Aij) indicates whether there is a relationship between node (i) and node (j). The node feature matrix (X) contains important information about each node in the graph, where each row of (X) represents the feature vector of a node [56].

GCNs use convolutional layers to aggregate information from neighboring nodes in the graph. These convolutional operations are designed to process the graph-structured data. In each layer, the features of each node are updated by aggregating features from neighboring nodes. This aggregation process is typically done using a weighted sum or a similar operation. During the training phase, the weights used for aggregation are learned.

Following the aggregation step, an activation function such as ReLU can be used to introduce nonlinearity into the model. Like Convolutional Neural Networks (CNNs), GCNs can also include pooling layers to subsample the graph. Pooling operations are performed to reduce the dimensionality of the graph while preserving important information. Common pooling strategies include max pooling and average pooling.

The final layer of the GCN produces the predictions or outputs of the model. Depending on the task, this layer may involve node classification, link prediction, or graph classification. GCNs are trained using labeled data like other neural networks. The model learns how to make predictions by minimizing a loss function, such as cross-entropy loss for classification tasks. Standard training involves backpropagation and gradient descent, where gradients for model parameters are computed (including weights and biases) and used to iteratively update the parameters.

An example of the graph convolutional network is mentioned in Figure 3.12. GCNs have demonstrated promising results in various applications, including social network analysis, recommendation systems, biological network analysis, and knowledge graph reasoning. Their ability to work directly on graph-structured data makes them suitable for tasks where relationships between elements play a crucial role [57].

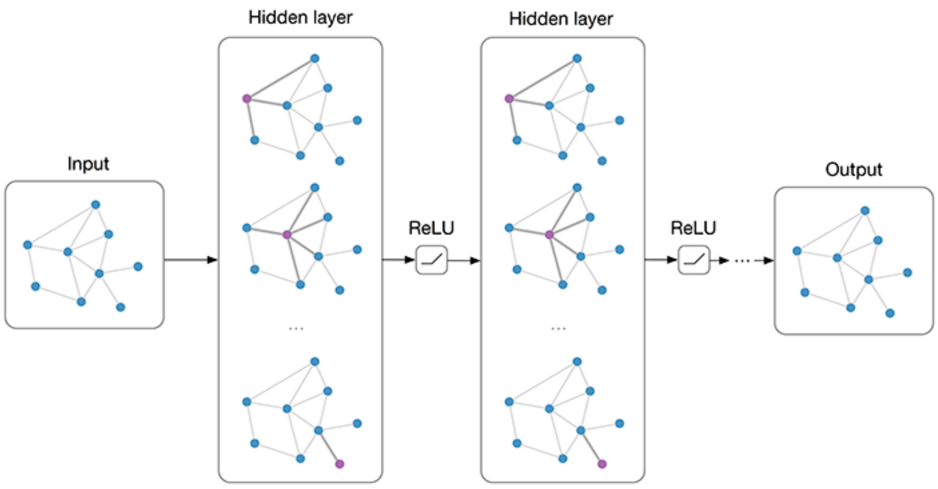


Figure 3.12 Sample Graph Convolutional Network Architecture [13]

3.3. Performance Parameters

This section provides an overview of the performance metrics used to evaluate machine learning models. These metrics are computed using four fundamental elements: True Positive (TP), True Negative (TN), False Positive (FP), and False Negative (FN).

Accuracy Score

The Accuracy Score is a measure that quantifies the overall correctness of the model's predictions. It is computed as the ratio of correct predictions (both positive and negative) to the total number of predictions. The formula for this calculation is depicted in Equation 4.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{4}$$

Precision Score

The Precision Score is a measure that quantifies the correctness of the model's positive predictions. It is computed as the ratio of true positive predictions to the total number of positive predictions (both true and false). The formula for this calculation is depicted in Equation 5.

$$Precision = \frac{TP}{TP + FP} \tag{5}$$

Recall Score

The Recall Score, also known as sensitivity, is a measure that quantifies the model's ability to correctly identify positive instances. The formula for this calculation is depicted in Equation 6.

$$Recall = \frac{TP}{TP + FN} \tag{6}$$

F1 Score

The F1 Score is a measure that balances the precision and recall of the model. It is particularly useful when the data has imbalanced classes. The formula for this calculation is depicted in Equation 7.

$$F1 = \frac{2 * Precision * Recall}{Precision + Recall} = \frac{2 * TP}{2 * TP + FP + FN} \tag{7}$$

Specificity

The Specificity Score, also known as the true negative rate, is a measure that quantifies the model's ability to correctly identify negative instances. The formula for this calculation is depicted in Equation 8.

$$Specificity = \frac{TN}{FP + TN} \tag{8}$$

ROC-AUC

The ROC-AUC Score is a comprehensive performance metric for classification models. It represents the probability that the model ranks a random positive instance higher than a random negative instance. It is computed as the area under the Receiver Operating Characteristic (ROC) curve, which plots the true positive rate against the false positive rate. The formulas for calculating the true positive rate is mentioned in Equation 9. False positive rate calculation formula is mentioned in Equation 10. Examples of the ROC curve and the area under the curve, are shown in Figure 3.13 [14].

$$TPR = \frac{TP}{TP + FN} \tag{9}$$

$$FPR = \frac{FP}{FP + TN} \tag{10}$$

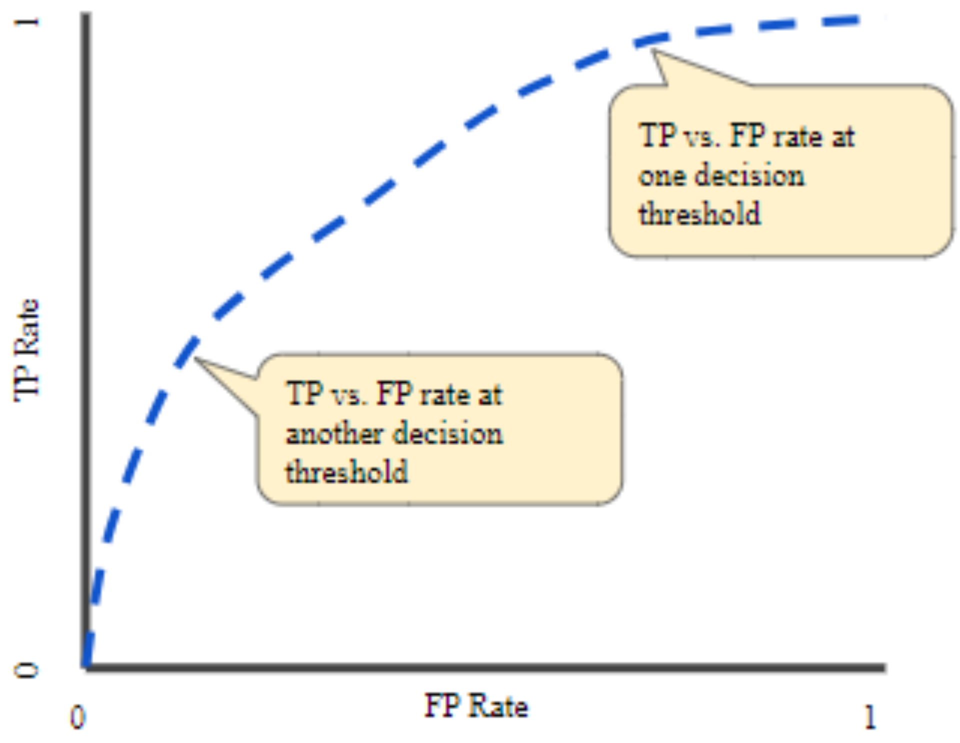


Figure 3.13 Example ROC Curve[14]

PR-AUC

The PR-AUC Score is another comprehensive performance metric for classification models. It represents the area under the Precision-Recall curve, which is particularly useful when dealing with imbalanced datasets. An example of a PR-AUC curve is shown in Figure 3.14 [15].

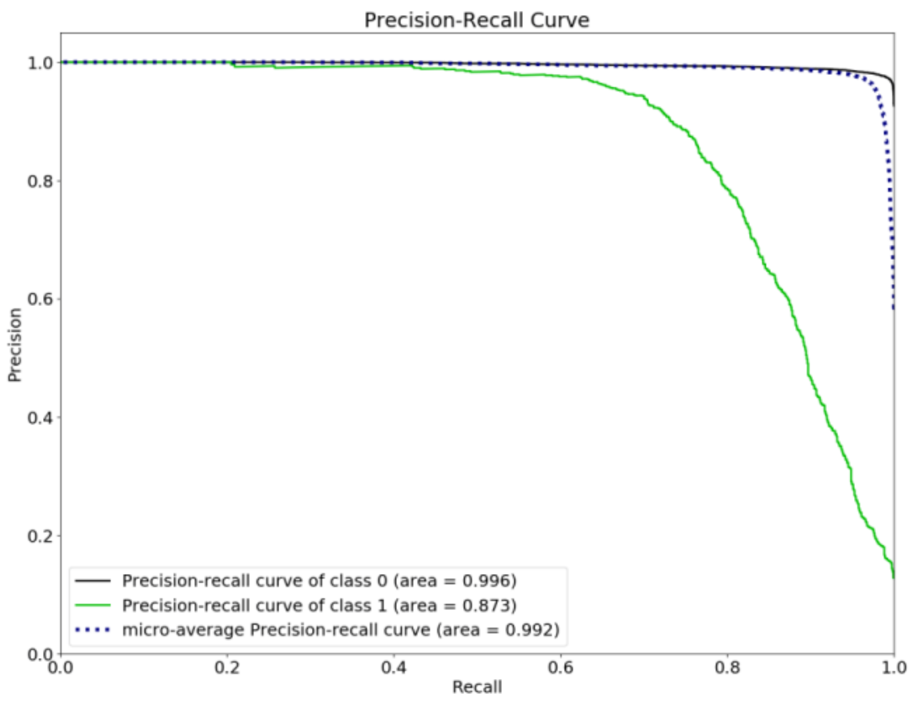


Figure 3.14 Example PR-AUC Curve [15]

MRR Score

The MRR Score, or Mean Reciprocal Rank, is a statistical measure used in information retrieval that quantifies the ability of a model to rank relevant items highly. It is calculated using the predicted output array of a machine learning model, where the position of the actual value in the array plays a key role. The formula for this calculation is depicted in Equation 11 [58].

$$MRR = \frac{1}{|Q|} \sum_{i=1}^{|Q|} \frac{1}{rank_i} \tag{11}$$

4. RELATED WORK

4.1. Graph Interactions

This section discusses various methods used to predict associations between two nodes. These methods primarily leverage the interaction of graphs to predict interactions within the graph.

One such method is proposed by Huang et al. [59]. They introduce a graph neural network that predicts molecular interactions by combining both direct similarity and skipped similarity interactions. This approach has proven useful over the past decade in various interacting networks, particularly in biological networks where similarity between non-interacting nodes is often significant. The proposed network, known as SkipGNN, receives messages from direct and two-hop neighbors of the interacting network and transforms them to obtain useful information for prediction. To incorporate hop similarity, a hop graph is created which is a modified version of the original network. An iterative aggregation scheme is developed to optimize the GNN using both the jump graph and the seed graph.

Block schema of SkipGNN [59], starts by constructing a skip graph from the input graph using an adjacency matrix and a specific equation. Initial node embeddings are determined using side information or generated using node2vec, and these embeddings are propagated along the edges of the graph. Through an iterative fusion process involving multiple layers and weight matrices, powerful embeddings are produced. These embeddings are then used in a decoder module that calculates the probability of interaction between nodes. The architecture aims to incorporate additional structural information to improve prediction accuracy for interactions within a graph.

SkipGNN [59] has been tested on four interactive networks, namely protein-protein graph, disease-gene graph, drug-target graph, and drug-drug graph. The results show that SkipGNN performs well, learning meaningful embeddings and demonstrating superior performance on noisy and incomplete networks.

SkipGNN [59] was tested on drug-target interaction graph, drug-drug interaction graph, protein-protein interaction graph and disease-gene interaction graph. In drug-target interaction graph, SkipGNN [59] performance result are 0.928(PR-AUC) and 0.922(ROC-AUC). In drug-drug interaction graph, SkipGNN [59] performance result are 0.866(PR-AUC) and 0.886(ROC-AUC). In protein-protein interaction graph, SkipGNN [59] performance result are 0.921(PR-AUC) and 0.917(ROC-AUC). In disease-gene interaction graph, SkipGNN [59] performance result are 0.915(PR-AUC) and 0.912(ROC-AUC).

Another research, HOGCN [60], focuses on gathering feature representations from neighbors at different distances and learning their linear mixture to obtain informative representations of biomedical entities. The higher power indicates a higher-order neighborhood for the training process. HOGCN performs accurate predictions on interaction networks, such as protein-protein, drug-drug, drug-target, and disease-gene networks. It proves effective on noisy and sparse networks by considering characteristic manifestations of neighbors at different distances. Additionally, the new interaction predictions are validated by literature-based case studies.

The HOGCN [60] block schmea begins with a graph of nodes connected by edges, which undergoes the Higher Order Graph Convolution (HOGC) process. This step enhances the graph's structure, making certain features more prominent. The processed graph is then subjected to bilinear operations, followed by linear transformations, resulting in the final output that likely represents the probability of interaction between nodes, denoted as "pij". This methodology is part of the broader field of machine learning, where such networks are used to analyze and predict complex relationships within data.

HOGCN[60] was tested on drug-target interaction graph, drug-drug interaction graph, protein-protein interaction graph and disease-gene interaction graph. In drug-target interaction graph, HOGCN[60] performance result are 0.937(PR-AUC) and 0.934(ROC-AUC). In drug-drug interaction graph, HOGCN[60] performance result are

0.897(PR-AUC) and 0.911(ROC-AUC). In protein-protein interaction graph, HOGCN[60] performance result are 0.93(PR-AUC) and 0.922(ROC-AUC). In disease-gene interaction graph, HOGCN[60] performance result are 0.941(PR-AUC) and 0.936(ROC-AUC).

In the work of Zenheng et al. [61], the Residual Message Graph Association Network (ResMGCN) is introduced as a method for the swift and precise prediction of biomedical interactions. ResMGCN distinguishes itself by aggregating lower-order information with higher-order information from the subsequent round to guide the node update and obtain a more meaningful node representation. It is capable of recognizing and retaining various messages from the previous layer and higher-order information in the current layer, using lower memory and time overhead to obtain the information representations of biomedical entities.

ResMGCN [61] was tested on drug-target interaction graph, drug-drug interaction graph, protein-protein interaction graph and disease-gene interaction graph. In drug-target interaction graph, ResMGCN[61] performance result are 0.918(PR-AUC) and 0.901(ROC-AUC). In drug-drug interaction graph, ResMGCN[61] performance result are 0.931(PR-AUC) and 0.936(ROC-AUC). In protein-protein interaction graph, ResMGCN [61] performance result are 0.91(PR-AUC) and 0.894(ROC-AUC). In disease-gene interaction graph, ResMGCN [61] performance result are 0.935(PR-AUC) and 0.925(ROC-AUC).

Kipf et al. [62] present(GCN) a scalable approach for semi-supervised learning on graph-structured data. This approach is based on an efficient variant of convolutional neural networks that operate directly on graphs. The model can scale linearly with the number of edges of the graph and learns hidden layer representations that encode both the local graph structure and node characteristics.

GCN was tested on drug-target interaction graph, drug-drug interaction graph, protein-protein interaction graph and disease-gene interaction graph. In drug-target interaction graph, GCN performance result are 0.904(PR-AUC) and 0.899(ROC-AUC) [59]. In drug-drug interaction graph, GCN performance result are 0.856(PR-AUC) and

0.877(ROC-AUC) [59]. In protein-protein interaction graph, GCN performance result are 0.909(PR-AUC) and 0.907(ROC-AUC) [59]. In disease-gene interaction graph, GCN performance result are 0.909(PR-AUC) and 0.906(ROC-AUC) [59].

In another work, Kipf et al [63] introduce the Variational Graph Autoencoder (VGAE), a framework for unsupervised learning on graph-structured data based on the Variational Autoencoder (VAE). The VGAE model is capable of learning explainable latent representations for undirected graphs and achieves competitive results in the task of link prediction in citation networks.

VGAE[63] was tested on drug-target interaction graph, drug-drug interaction graph, protein-protein interaction graph and disease-gene interaction graph. In drug-target interaction graph, VGAE performance result are 0.853(PR-AUC) and 0.8(ROC-AUC) [59]. In drug-drug interaction graph, VGAE performance result are 0.844(PR-AUC) and 0.878(ROC-AUC) [59]. In protein-protein interaction graph, VGAE performance result are 0.875(PR-AUC) and 0.844(ROC-AUC) [59]. In disease-gene interaction graph, VGAE performance result are 0.902(PR-AUC) and 0.873(ROC-AUC) [59].

Xu et al. [64] presents(GIN) a theoretical framework to analyze the expressive ability of Graph Neural Networks (GNNs) to capture different graph structures. The proposed method, developed based on a simple architecture, is perhaps the most expressive among the GNN classes and is as powerful as the Weisfeiler-Lehman graph isomorphism test.

GIN[64] was tested on drug-target interaction graph, drug-drug interaction graph, protein-protein interaction graph and disease-gene interaction graph. In drug-target interaction graph, GIN performance result are 0.853(PR-AUC) and 0.8(ROC-AUC) [59]. In drug-drug interaction graph, GIN [64] performance result are 0.844(PR-AUC) and 0.878(ROC-AUC).[59] In protein-protein interaction graph, GIN performance result are 0.875(PR-AUC) and 0.844(ROC-AUC) [59]. In disease-gene interaction graph, GIN performance result are 0.902(PR-AUC) and 0.873(ROC-AUC) [59].

Xu et al. [65] analyzes some important characteristics of recent deep learning methods for learning graph representations(JK-NET). These methods follow a neighborhood aggregation procedure and the proposed algorithm overcomes other methods. An architecture, known as the knowledge network, is explored to adapt to the properties and tasks of the local neighborhood.

JK-NET[65] was tested on drug-target interaction graph, drug-drug interaction graph, protein-protein interaction graph and disease-gene interaction graph. In drug-target interaction graph, JK-NET performance result are 0.921(PR-AUC) and 0.907(ROC-AUC) [59]. In drug-drug interaction graph, JK-NET performance result are 0.870(PR-AUC) and 0.885(ROC-AUC) [59]. In protein-protein interaction graph, JK-NET performance result are 0.912(PR-AUC) and 0.901(ROC-AUC) [59]. In disease-gene interaction graph, JK-NET performance result are 0.891(PR-AUC) and 0.898(ROC-AUC) [59].

Abu-El-Haija et al. [66], it is demonstrated that existing popular semi-supervised learning methods with GNN and GCN cannot learn a general class of mixed neighborhood relations. A new model is proposed to handle this weakness. This model uses mixing the feature representations of the neighbors repeatedly, approached at different distances.

The MixHop [66] block schema compares two methods of feature propagation in graph convolution networks. The traditional method aggregates features only from immediate neighbors, while the MixHop model allows for the aggregation of features from neighbors at multiple distances. This enables the central node, highlighted in red, to learn a combination of features from various layers of neighbors, enhancing the model's ability to capture and utilize a richer set of information for analysis or prediction tasks in network analysis and machine learning. The MixHop approach represents a significant advancement in the field, offering a more nuanced and comprehensive way to understand and leverage the complex relationships within data.

MixHop[66] was tested on drug-target interaction graph, drug-drug interaction graph, protein-protein interaction graph and disease-gene interaction graph. In drug-target interaction graph, MixHop performance result are 0.921(PR-AUC) and 0.92(ROC-AUC)

[59]. In drug-drug interaction graph, MixHop performance result are 0.861(PR-AUC) and 0.879(ROC-AUC) [59]. In protein-protein interaction graph, MixHop performance result are 0.909(PR-AUC) and 0.913(ROC-AUC) [59]. In disease-gene interaction graph, MixHop performance result are 0.912(PR-AUC) and 0.916(ROC-AUC) [59].

4.2. Network Similarity

This section discusses research that utilizes the neighbors of nodes to predict interactions within the graph.

In the work of Kovács et al. [67], the L3 heuristic is introduced. Recent developments have shown that the common neighbor hypothesis is not effective for most protein pairs in Protein-Protein Interaction (PPI) prediction. The L3 heuristic proposes to consider nodes that are similar to a node's neighbors, quantified with A^3, where A represents the adjacency matrix. This suggests that higher-order neighbors play a significant role in predicting interactions.

The neighbor algorithm of the L3, a length-3 path is a sequence of three interactions that link two proteins indirectly. For example, if protein D is connected to protein Y through a series of three other proteins, this constitutes one length-3 path. The algorithm calculates the connection probability by considering all such length-3 paths between the protein pair.

L3[67] was tested on drug-target interaction graph, drug-drug interaction graph, protein-protein interaction graph and disease-gene interaction graph. In drug-target interaction graph, L3 performance result are 0.891(PR-AUC) and 0.793(ROC-AUC) [59]. In drug-drug interaction graph, L3 performance result are 0.86(PR-AUC) and 0.869(ROC-AUC) [59]. In protein-protein interaction graph, L3 performance result are 0.899(PR-AUC) and 0.861(ROC-AUC) [59]. In disease-gene interaction graph, L3 performance result are 0.899(PR-AUC) and 0.832(ROC-AUC) [59].

Tang et al. [68] propose a framework that initially extracts social aspects based on network structure to accurately capture crucial interaction patterns between agents. Subsequently, a discriminative classifier is learned to select the relevant societal aspects.

4.3. Network Embedding

This section discusses the utilization of graph node embeddings to predict interactions within the graph.

Perozzi et al. [69] introduced a novel approach for learning latent representations of vertices in networks, thereby encoding social relationships into a vector space suitable for statistical models. DeepWalk, by leveraging truncated random walks, extends the concepts of language modeling and unsupervised feature learning to graphs. It provides perspectives on multi-label network classification tasks for various social networks and other types of networks.

Deepwalk[69] was tested on drug-target interaction graph, drug-drug interaction graph, protein-protein interaction graph and disease-gene interaction graph. In drug-target interaction graph, Deepwalk performance result are 0.753(PR-AUC) and 0.735(ROC-AUC) [59]. In drug-drug interaction graph, Deepwalk performance result are 0.698(PR-AUC) and 0.712(ROC-AUC) [59]. In protein-protein interaction graph, Deepwalk performance result are 0.715(PR-AUC) and 0.706(ROC-AUC) [59]. In disease-gene interaction graph, Deepwalk performance result are 0.827(PR-AUC) and 0.832(ROC-AUC) [59].

Grover et al. [70] proposed node2vec, an algorithmic framework for learning continuous feature representations for nodes in a network. Node2vec discovers a mapping of nodes to a low-dimensional feature space that optimizes the preservation of network neighborhoods. The flexibility in defining a node's network neighborhood and the design of a biased random walk process allows for effective exploration of different neighborhoods. This algorithm generalizes previous work based on rigid notions of network neighborhoods, arguing that flexibility in neighborhood discovery is crucial for richer performance learning.

Node2vec[70] was tested on drug-target interaction graph, drug-drug interaction graph, protein-protein interaction graph and disease-gene interaction graph. In drug-target interaction graph, Node2vec performance result are 0.771(PR-AUC) and 0.72(ROC-AUC) [59]. In drug-drug interaction graph, Node2vec performance result are 0.801(PR-AUC) and 0.809(ROC-AUC) [59]. In protein-protein interaction graph, Node2vec performance result are 0.773(PR-AUC) and 0.766(ROC-AUC) [59]. In disease-gene interaction graph, Node2vec performance result are 0.828(PR-AUC) and 0.834(ROC-AUC) [59].

Riberio et al. [71] presented struc2vec, a flexible framework for learning latent representations for node structure recognition. Struc2vec measures the similarity of nodes at different scales and constructs a multi-layer graph to encode structural similarities and create structural context for nodes.

Struc2vec[71] was tested on drug-target interaction graph, drug-drug interaction graph, protein-protein interaction graph and disease-gene interaction graph. In drug-target interaction graph, Struc2vec performance result are 0.677(PR-AUC) and 0.656(ROC-AUC) [59]. In drug-drug interaction graph, Struc2vec performance result are 0.643(PR-AUC) and 0.654(ROC-AUC) [59]. In protein-protein interaction graph, Struc2vec performance result are 0.875(PR-AUC) and 0.868(ROC-AUC) [59]. In disease-gene interaction graph, Struc2vec performance result are 0.91(PR-AUC) and 0.909(ROC-AUC) [59].

5. METHODS

GLADIGATOR (Graph Learning-bAsed DIsease Gene AssociaTiOn pRediction) is a novel computational framework designed to enhance the prediction of disease-gene associations. Leveraging the power of graph learning, GLADIGATOR aims to uncover the complex relationships between genes and diseases. The method integrates heterogeneous biological data sources into a unified graph structure, where nodes represent genes and diseases, and edges signify known associations. By applying advanced graph learning algorithms, GLADIGATOR can learn the intricate patterns within the graph, enabling it to predict potential disease-gene associations. This approach not only facilitates a deeper understanding of genetic influences on diseases but also aids in the identification of new therapeutic targets, thereby accelerating the pace of medical research and drug development.

5.1. Data

The data is procured from DisGeNET (version 7.0). The database encompasses 369,554 variant-disease associations (VDAs), associating 194,515 variants with 14,155 diseases, features, and phenotypes. Additionally, it includes 1,134,942 gene disease associations (GDAs), linking 21,671 genes to 30,170 diseases, disorders, traits, and clinical or abnormal human phenotypes [3].

5.1.1. Dataset Attributes

The DisGeNET database amalgamates human gene-disease associations (GDAs) and variant-disease associations (VDAs) data from a multitude of sources, including Mendelian, complex, and environmental disorders. The integration is facilitated by the DisGeNET association type ontology and gene and disease vocabulary mapping. In this section provides a detailed explanation of the attributes of genes, diseases, gene-disease associations, and disease-disease associations.

5.1.1.1. Gene

The DisGeNet database contains the following attributes for genes [3]:

- **Gene Symbol**: The gene symbol is obtained from NCBI.
- **Gene Name**: The gene name is also sourced from NCBI.
- **UniProtID**: The UniProt database, a comprehensive resource for protein sequence and functional data, assigns unique identifiers to individual protein sequences called UniProtIDs. It is used to make sure that every protein sequence is unique and able to be cited in scientific publications and research. It is a component of the UniProt Knowledgebase (UniProtKB).[72]
- **Disease Specificity Index (DSI)**: This parameter is used to determine whether a gene is associated with a specific disease or a group of diseases. The formula for calculating the DSI is shown in Equation 12. In this formula, N_d represents the number of diseases associated with the gene, and N_T is the total number of diseases in DisGeNet.

$$DSI = \frac{log_2(\frac{N_d}{N_t})}{log_2(\frac{1}{N_t})} \tag{12}$$

- **Disease Pleiotropy Index (DPI)**: This parameter is used to understand whether a gene is associated with one or more MeSH disease classes. The DPI score can help researchers easily understand this. The DPI calculation formula is shown in Equation 13. In this formula, N_{dc} represents the number of disease classes associated with the gene, and N_{TC} is the total number of disease classes in DisGeNet.

$$DPI = (\frac{N_{dc}}{N_{TC}}) * 100 \tag{13}$$

- **PLI**: The PLI (pLI score) is used to indicate the likelihood that the gene is intolerant to loss-of-function mutations.

5.1.1.2. Disease

The DisGeNet database provides the following attributes for diseases [3]:

- **Disease Name**: The name of the disease is sourced from UMLS.
- **Disease Semantic Type**: The semantic type of the disease is obtained from UMLS semantic types.
- **Disease Class**: This refers to the MeSH class of the disease.
- **Disease Type**: This refers to the types of diseases as classified by DisGeNet.

5.1.1.3. Gene-Disease Association

The sources of gene disease association are divided into four categories:

- **Curated Data** : The Curated Data is constructed based on several databases such as ClinGen, CTD, CGI, Orphanet, Genomics England PanelApp, PsyGeNET, and UniProt [72–76].
- **Animal Models Data** : The Animal Models Data is built on GDAs from RGD, MGD, and CTD databases [74, 77, 78].
- **Inferred Data** : The Inferred Data is constructed based on HPO, Clinvar, GWAS Catalog, and GWAS DB [79–82].
- **Literature Data** : The Literature Data is built on LHGDN and BeFree [83–86].

The sources of GDA associations are illustrated in Table 5.1. The protein class distributions associated with genes in the database are presented in Figure 5.1. The sources of GDAs in the database are depicted in Figure 5.2, where the Venn diagram explains the overlap between the sources.

Source	Genes	Diseases	Assocs	Evidence
CGI	315	200	1557	1557
CLINGEN	634	447	1260	7858
GENOMICS ENGLAND	3967	6046	11215	18542
CTD_human	8247	8246	67471	84380
ORPHANET	3356	3266	6398	8322
PYSGENET	1393	105	3290	6728
UNIPROT	3894	3935	5728	17564
CURATED	**9703**	**11181**	**84038**	**137822**
HPO	4281	7591	164198	164198
CLINVAR	4467	9247	26002	85646
GWASDB	4862	450	11172	14663
GWASCAT	10403	948	40443	56795
INFERRED	**13258**	**14843**	**233738**	**313885**
CTD_mouse	70	292	475	518
CTD_rat	21	29	48	48
MGD	1776	2085	4598	8569
RGD	2143	1168	11667	13062
ANIMAL MODELS	**3334**	**3171**	**16660**	**22171**
LHGDN	5935	1793	31427	52794
BEFREE	18839	17993	846474	2700332
LITERATURE	**18898**	**18171**	**858354**	**2738700**
ALL	***21671***	***30170***	***1134942***	***3178358***

Table 5.1 DisGeNet GDA Associations Sources Statistics [3]

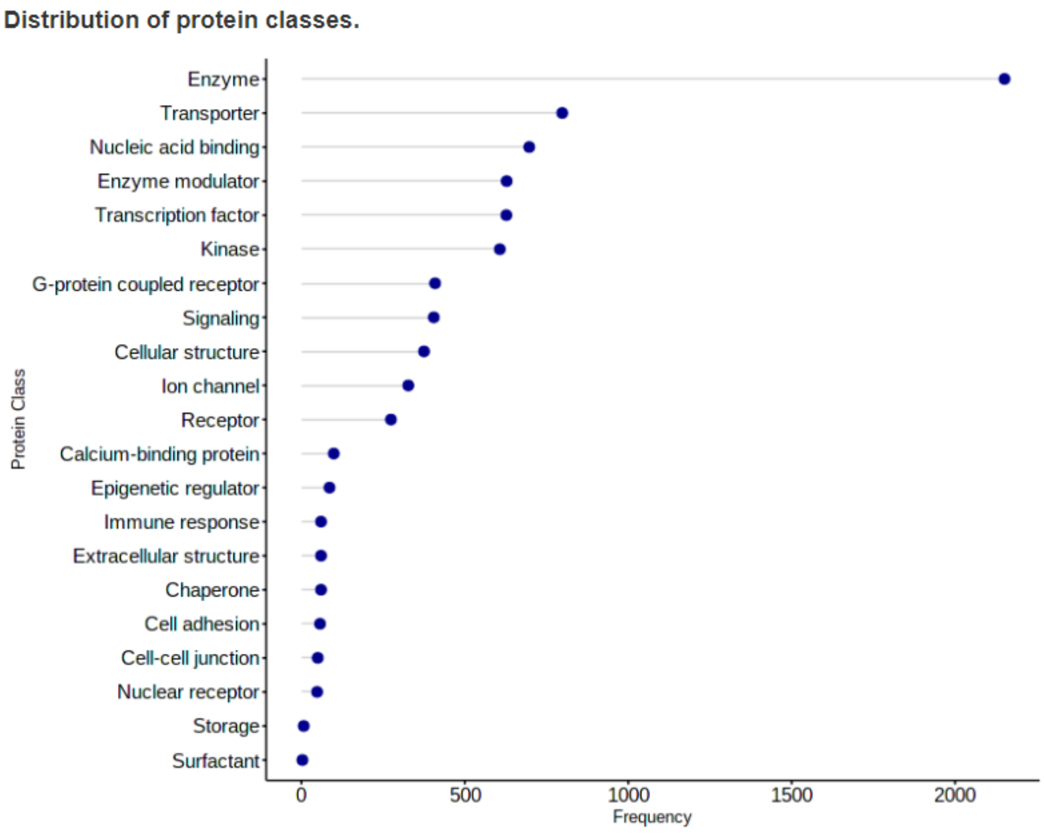


Figure 5.1 Protein Class Distribution [3]

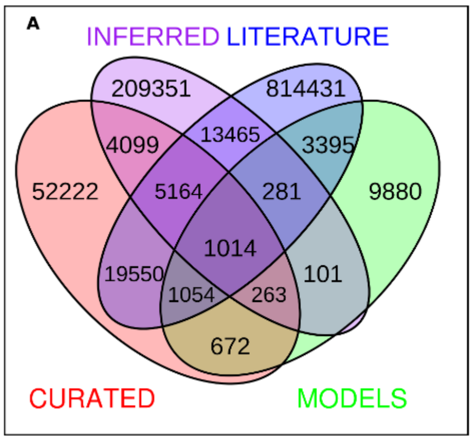


Figure 5.2 GDAs' Sources Venn Schema [3]

All disease-gene associations in the database have a score parameter that indicates the reliability of the association. The formula for calculating the gene-disease association score is shown in Equation 14.

$$S = C + M + I + L \tag{14}$$

The score calculation formula (Equation 14) consists of four main parameters: C, M, I, and L. The explanations and formulas for these parameters are as follows [3]:

- **C**: This parameter represents a formula that uses the number of sources containing this gene disease association. The sources include CGI, CLINGEN, GENOMICS ENGLAND, CTD, PSYGENET, ORPHANET, and UNIPROT [72–76, 81]. The formula is shown in Equation 15 [3].

$$C = \begin{cases} 0.6 & if N_{sources_i} > 2 \\ 0.5 & if N_{sources_i} = 2 \\ 0.3 & if N_{sources_i} = 1 \\ 0 & otherwise \end{cases} \tag{15}$$

- **M**: This parameter represents a formula that uses the number of sources containing this gene disease association. The sources include RGD, MGD, and CTD [74, 77, 78]. The formula is shown in Equation 16 [3].

$$M = \begin{cases} 0.2 & if N_{sources_j} > 0 \\ 0.0 & otherwise \end{cases} \tag{16}$$

- **I**: This parameter represents a formula that uses the number of sources containing this gene disease association. The sources include HPO, CLINVAR, GWASCAT, and GWASDB [79–82]. The formula is shown in Equation 17 [3].

$$I = \begin{cases} 0.1 & if N_{sources_k} > 0 \\ 0.0 & otherwise \end{cases} \tag{17}$$

- **L**: This parameter represents a formula that uses the number of publications containing this gene disease association. The sources include LHGDN and BEFREE [83–86]. The formula is shown in Equation 18 [3].

$$L = \begin{cases} 0.1 & if N_{pubs} > 9 \\ N_{pubs} * 0.01 & N_{pubs} <= 9 \end{cases} \tag{18}$$

The GDA score calculation formula is a composite formula that includes other formulas. In DisGeNet, GDA scores are clustered. This statistic is shown in Figure 5.3.

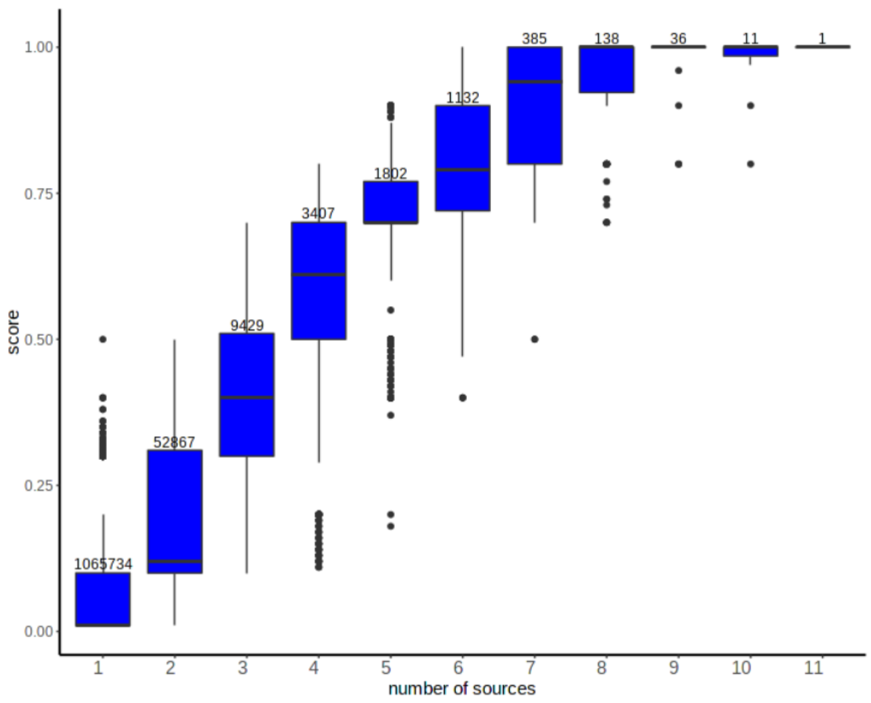


Figure 5.3 Number of GDA Sources Distrubution[3]

The DisGeNet database provides the following attributes for disease-gene associations [3]:

- **DisGeNet Score**: This is the score assigned to the disease-gene association.

- **Association Type**: This refers to the type of disease-gene association.

- **Evidence Level**: The evidence level, created by ClinGen[73], measures the extent of evidence supporting a disease-gene association.
- **Evidence Index**: This parameter is a measure of reliability. The calculation formula for the evidence index is shown in Equation 19. In this formula, $N_{pubs_{positive}}$ represents the number of publications that contain this disease-gene association in BeFree[85],[86] or PsyGeNET[76], and $N_{pubs_{total}}$ is the total number of publications in BeFree[85],[86] or PsyGeNET[76].

$$EI = \frac{N_{pubs_{positive}}}{N_{pubs_{total}}} \tag{19}$$

- **First Publish Year**: This is the year when the first publication was published.
- **Last Publish Year**: This is the year when the last publication was published.
- **Publication ID**: This is the ID of the publication from PubMed.
- **Source**: This is the source where the association was reported.

5.1.1.4. Disease-Disease Association

The DisGeNet database provides the following attributes for disease-disease associations [3]:

- **Jaccard Index**: This parameter measures the similarity between diseases based on their related genes. The calculation formula for the Jaccard index is shown in Equation 20. In this formula, G_1 represents the number of genes related to disease D_1, and G_2 represents the number of genes related to disease D_2 [3].

$$Jaccard_G = \frac{G_1 \cap G_2}{G_1 \cup G_2} \tag{20}$$

- **p-value**: This is the p-value of the association.

5.2. Features

5.2.1. Gene/Protein Features

This section elaborates on gene features. Each gene is associated with a specific protein, and hence, the attributes of these proteins can be utilized. Therefore, protein features can be used. Embeddings are generated from features, offer computational tools that allow computers to easily understand.

Protein embeddings are generated from the structural and functional properties of a protein, often based solely on its sequence. Despite the high computational cost of generating these embeddings, they can be utilized in various tasks such as sequence classification, sequence grouping, and sequence similarity search once calculated.

Machine learning models are used to generate vectors that computers can easily understand. Prot-T5[1] is a popular example of these methods. UniprotKB has generated protein embedding with Prot-T5[1] for every protein and published the data. These methods can use protein sequence and other features. Prot-T5[1] method uses protein sequence feature. The output vector size of Prot-T5[1] is (1*1024).

Prot-T5[1] is part of the ProtTrans project and represents a significant advancement in the field of bioinformatics, specifically in the understanding of protein sequences. It is a protein language model that has been trained on a vast dataset containing billions of amino acids. This training allows Prot-T5 to predict various aspects of protein function and structure with high accuracy.

The model itself is built upon the T5 architecture[1], which stands for Text-to-Text Transfer Transformer. This architecture includes both an encoder and a decoder, making it highly effective for tasks that involve translating one sequence into another. In the context of proteins, this means interpreting the sequences of amino acids—the building blocks of proteins—and translating them into a form that can be used to predict their properties and functions.

The steps for selecting gene/protein features are outlined below. The entire process is summarized in Figure 5.4

- **Gene Feature Selection** :
  Protein sequences have been chosen as gene features. Given that each protein sequence is unique to its related genes, it serves as a distinguishing feature.
- **Gene Feature Generator Methods** :
  Protein sequences have been chosen as gene features. Among the various approaches, machine learning-based methods are the most popular. Hence, a machine learning-based method has been selected for this project. The Prot-T5[1] method, which is quite popular, is used in this project. The output size of this method is (1*1024).

5.2.2. Disease Features

This section explains disease features. The focus of this project is on diseases that are related to genes. Disease embeddings are genrated from disease fetures, provide computational tools that allow computers to easily understand (in vector representation) the structural and functional properties of a disease. Diseases do not have specific features like protein sequence. Instead, they have features such as disease type, disease class, disease name, disease description, etc. The operation of generating disease embedding can be built on NLP methods.

NLP methods are generic methods that can be used in this context. Examples of NLP methods include Tf-Idf, IDF, tokenization, word embeddings, etc. These methods should utilize features such as disease name, disease description, etc., which are text-based features. Therefore, disease name and disease description can be used to generate disease embeddings [87].

Machine learning models are used to generate vectors that computers can easily understand. BioBert[2] is a popular example of these methods. BioBert[2] is a pre-trained machine learning model that generates vectors whose input is biological context. BioBert[2] can be used with disease name and disease description. BioBert[2], a pre-trained machine learning model, generates (1*768) sized vectors.

Disease class and disease type are among the features of diseases. These features are useful. However, these features are not disease-specific features. Therefore, these features are useful for assisting the model with other features.

The selection of disease features is a crucial step in the process. The entire process is depicted in Figure 5.4

- **Disease Feature Selection** :
  The features chosen for diseases are the disease name and description. Each disease has a unique name and description, making them ideal discriminative features. In this project, a combination of the disease name and description is used.
- **Refinement** :
  The disease name and description are chosen as inputs for the vector generator algorithms. These features need to be refined, which includes steps like removing English stop words and punctuations.
- **Disease Feature Generator Methods** :
  The disease definition, a text-based information, is chosen as a disease feature. Machine learning based methods are popular approaches for this purpose. In this project, the BioBert[2] method, a widely used method, is employed. The output size of this method is (1*768).

5.2.3. Aggregation

Protein sequences are used for genes, while the disease name and description are used for diseases. The vector sizes for genes and diseases are (1*1024) and (1*768) respectively. Given the difference in vector sizes, adjustments need to be made. The main reason is that the gene embedding and disease embedding are generated different vector spaces. The steps for this solution are outlined below, and the entire aggregation process is summarized in Figure 5.4

- **Gene Vector Customization** :
  To standardize the vector size, zeros are added to the end of the vector. Specifically, 768 zeros are appended to the generated vector to achieve a vector size of (1*1792).
- **Disease Vector** :
  To standardize the vector size, zeros are added to the beginning of the vector. Specifically, 1024 zeros are prepended to the generated vector to achieve a vector size of (1*1792).

5.3. Graph Preparation for the Model

5.3.1. Nodes

Nodes are the basic units of various data structures, including linked lists and tree data structures. They hold data and have the ability to link to other nodes. Pointers are often used to establish connections between nodes.

Information about genes and diseases is stored in nodes. Each node has three parameters: **x**, **id**, and **gene_symbol**. These parameters are explained below.

- **x** :
  This parameter stores the embeddings of proteins or diseases. The size of the embedding is (1*1792).
- **id** :
  This parameter stores the id information of the protein or disease.

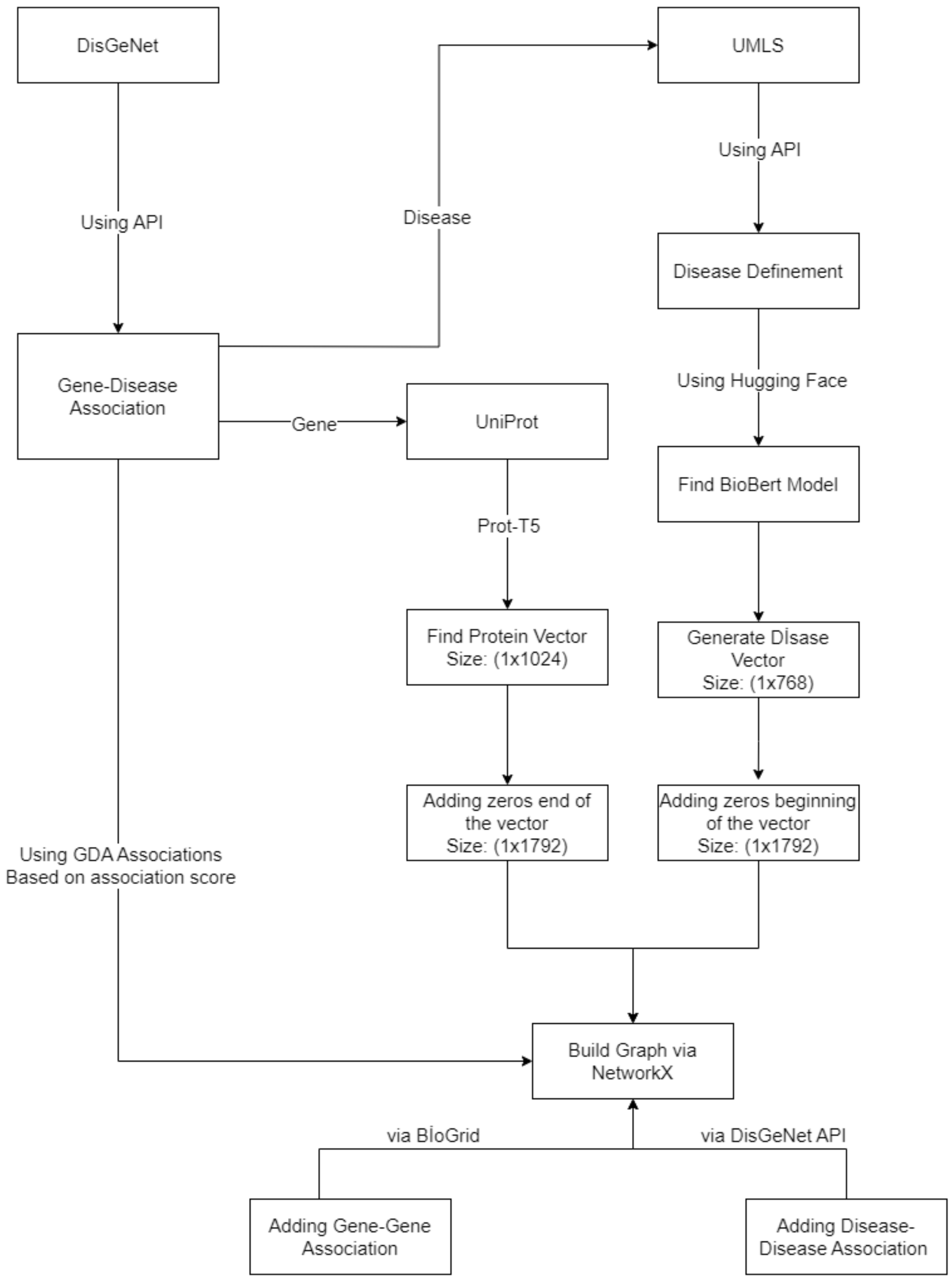

DisGeNet
Using API
Disease
UMLS
Using API
Disease Definement
Gene-Disease Association
Gene
UniProt
Using Hugging Face
Find BioBert Model
Prot-T5
Find Protein Vector Size: (1x1024)
Generate Dİsase Vector Size: (1x768)
Adding zeros end of the vector Size: (1x1792)
Adding zeros beginning of the vector Size: (1x1792)
Using GDA Associations Based on association score
Build Graph via NetworkX
via BİoGrid
via DisGeNet API
Adding Gene-Gene Association
Adding Disease-Disease Association

Figure 5.4 Graph Construction Scenario

- **gene_symbol** :

  This parameter stores the gene symbol information of the nodes. If the node contains gene information, the **gene_symbol** parameter is not null. However, if the node contains disease information, the **gene_symbol** parameter is null.

5.3.2. Edges

An edge, also known as a link, is a connection established by one node (or vertex) to another node (or vertex). The edges of a node can be positioned to point in various directions.

Edges are created between two nodes. Each node is assigned to a gene or disease. Therefore, each edge has an **edge_nodes_attributes** parameter. This parameter is explained below.

- **edge_nodes_attributes** :
  This parameter is stored in every edge. It contains information about the start and stop nodes of the edge. This information is built based on id identifiers.

5.3.3. Graph Construction

5.3.3.1. GDA Score Based Main Graph Creation

The algorithm for building a GDA score based main graph is mentioned below. This graph uses data gathered from the DisGeNet database API[3]. All these steps are depicted in Figure 5.4

- **Data from DisGeNet API** :
  Information about genes, diseases, disease-gene associations, and disease-disease associations is retrieved from the DisGeNet API.
- **Data from BioGrid** :
  Information about gene-gene associations in Homo sapiens (humans) is gathered from BioGrid[88].

- **Clustering Edges** :
  The main edges are the disease-gene associations. The gene disease association score (DisGeNet Score) is a boundary condition used to cluster edges. For instance, if the gene disease association score is greater than or equal to 0.5, all gene disease associations with a score greater than or equal to 0.5 are clustered.

- **Adding Nodes to the Graph** :
  After the disease-gene associations are clustered, the main work area is constructed. Genes and diseases are gathered from the clustered disease-gene associations to add nodes to the graph. Each node represents a single gene or disease.

- **Adding Edges to the Graph** :
  After adding nodes, the steps for adding edges are mentioned below.

  - **Disease-Gene Edges** : Disease-gene edges are added based on the clustered disease-gene edges.
  - **Disease-Disease Edges** : Disease-disease edges are added based on the data gathered from the DisGeNet API[3].
  - **Gene-Gene Edges** : Gene-gene edges are added based on the data gathered from BioGrid[88].

5.3.3.2. Graph Construction for Comparison with SkipGNN

SkipGNN, a method for predicting molecular interactions, has been tested on drug-target interactions, drug-drug interactions, protein-protein interactions, and disease-gene interactions. In this project, we plan to make predictions on disease-gene associations. The methodologies of SkipGNN[59] and this research are comparable, necessitating the creation of a new graph-building algorithm.

The algorithm for building this graph is described below. The graph uses data gathered from the DisGeNet curated dataset[3]. All these steps are same in Figure 5.4, but the main difference is the source of the disease-gene association. Graph construction for comparison with SkipGNN's source is DisGeNet curated dataset.

- **Data from DisGeNet Curated Dataset** :
  The DisGeNet curated dataset provides disease-gene association data. This dataset is obtained from the SkipGNN research GitHub repository.
- **Data from BioGrid** :
  Information about gene-gene associations in Homo sapiens (humans) is gathered from BioGrid[88].
- **Adding Nodes to the Graph** :
  After the disease-gene associations are clustered, the main work area is constructed. Genes and diseases are gathered from the clustered disease-gene associations to add nodes to the graph. Each node represents a single gene or disease.
- **Adding Edges to the Graph** :
  After adding nodes, the steps for adding edges are mentioned below.
  - **Disease-Gene Edges** : Disease-gene edges are added based on the curated dataset.
  - **Disease-Disease Edges** : Disease-disease edges are added based on the data gathered from the DisGeNet API[3].
  - **Gene-Gene Edges** : Gene-gene edges are added based on the data gathered from BioGrid[88].

5.3.3.3. Graph Construction using OGB

The Open Graph Benchmark (OGB)[89] is a comprehensive database encompassing various datasets. The ogbl-biokg dataset within OGB, which includes associations between biological components, is utilized to construct an association prediction machine learning model. This dataset comprises disease-protein associations, drug-drug associations, and drug-protein associations. For the purpose of this research, protein-disease associations are isolated to construct a new graph, aiming to enhance the performance of the machine learning model.

The algorithm for constructing this graph is described below. The graph utilizes data gathered from the ogbl-biokg dataset(OGB). All these steps are same in Figure 5.4, but the main difference is the source of the disease-gene association. Graph Construction using OGB's source is ogbl-biokg dataset from OGB[89].

- **Data from ogbl-biokg(OGB)** :
  disease-gene associations are obtained from the ogbl-biokg dataset. This dataset is referenced in this link.
- **Data from BioGrid** :
  Information about gene-gene associations in Homo sapiens (humans) is gathered from BioGrid[88].
- **Adding Nodes to the Graph** :
  After the disease-gene associations are clustered, the main workspace is constructed. Genes and diseases are extracted from the clustered disease-gene associations to add nodes to the graph. Each node represents a single gene or disease.
- **Adding Edges to the Graph** :
  After adding nodes, the steps for adding edges are as follows:
  - **Disease-Gene Edges** : Disease-gene edges are added based on the ogbl-biokg dataset.
  - **Disease-Disease Edges** : Disease-disease edges are added based on the data gathered from the DisGeNet API[3].
  - **Gene-Gene Edges** : Gene-gene edges are added based on the data gathered from BioGrid[88].

5.3.4. Graph Transformation

The construction of the graph is executed using the NetworkX framework[90]. The structure of this graph is not compatible with machine learning models. Therefore, the graph is transformed into a PyTorch Geometric[91] graph, which is more suitable for machine learning models.

The dataset is transformed into a format that is more amenable to machine learning. The dataset stores the following parameters:

- **x** : A 2-dimensional data structure. The first dimension is the node index present in the graph. The second dimension is the embedding vector of the nodes.
- **edge_index** : A two-dimensional data structure that stores the identifiers of the starting and ending nodes of edges. The first dimension corresponds to the start node's identifier, while the second dimension contains the identifiers of the stop nodes.
- **id** : A 1-dimensional data structure that stores the id of the nodes.
- **gene_symbol** : A 1-dimensional data structure that stores the gene symbol of the node. If the node is a disease, the gene_symbol variable is null.
- **edge_node_attributes** : A 1-dimensional data structure that stores the start node id and stop node id. This variable is a text-based variable that concatenates 2 ids with a comma.

5.4. Constructed Graphs

5.4.1. Graph Identifiers

The constructed custom graphs need to be explicated. The explanations include source data, gene-disease score, etc. These details are provided in Table 5.2. Other tables do not contain the name of the graph or any related information except for the id variable. The id variables are replicated from Table 5.2 to other tables and references.

The first four graphs (Graphs 1-4) are constructed using data from the DisGeNet API, with each graph differentiated by the GDA score used. TThese graphs are contructed to analyze the impact of varying GDA scores on the model performance. Graph 5 is based on DisGeNet curated data, while Graph 6 utilizes data from ogbl-biokg (OGB). Different data sources are employed to construct these graphs, allowing for an assessment of the significance of each data source in the comparison. The last two graphs are designed to compare the performance of GLADIGATOR with other methods.

Graph ID	**Graph Name**	**Source Data**	**Gene-Disease Score**	Protein Embedding	Disease Embedding
1	Graph_Own_0.9	DisGeNet API	$0.9 \leq$	Prot-T5	BioBert
2	Graph_Own_0.5	DisGeNet API	$0.5 \leq$	Prot-T5	BioBert
3	Graph_Own_0.1	DisGeNet API	$0.1 \leq$	Prot-T5	BioBert
4	Graph_Own_0.05	DisGeNet API	$0.05 \leq$	Prot-T5	BioBert
5	Graph_Comparison_SkipGNN	DisGeNet curated	NA	Prot-T5	BioBert
6	Graph_Comparison_OGB	Open Graph Benchmark (only protein-disease part)	NA	Prot-T5	BioBert

Table 5.2 Constructed Graph Information

5.4.2. Graph Detailed Informatics

The customized graph's gene count, disease count, edge count, and other details are specified in Table 5.3. The identifiers for the graph are derived from Table 5.2. The smallest graph data is Graph 1, the biggest graph data is Graph 4. In first 5 graph, gene-gene association number is more than disease-disease association. However, disease-disease association number is more than gene-gene association number in Graph 6. Also, disease-disease association number is higher than disease-gene association number in only Graph 2.

Graph ID	Gene/Protein Number	Disease Number	Gene-Gene Association Number	Disease-Disease Association Number	Disease-Gene Association Number
1	741	836	3274	462	899
2	4188	5989	74180	11114	10152
3	13881	20481	494378	120409	335260
4	14185	21220	513255	127112	375609
5	8770	10231	269384	30063	78097
6	8581	9683	9058	27782	71064

Table 5.3 Detailed Information of Constructed Graph

5.5. Graph Partitioning

Partitioning the graph is a crucial step in the process. Machine learning models require different datasets for the training process. The partitioning operation aims to separate the data into different parts to train the machine learning model.

The data is categorized into three parts. One part is the training set, which is used to train machine learning models. Another part is the validation data, which is used to fine-tune the machine learning model. The last part of the data is the test data, which is used to evaluate the performance of the machine learning model.

The partitioning operation is a critical part of the machine learning model training process. Therefore, the steps of the partitioning operation are planned carefully. These partitioning operation steps are mentioned below.

5.5.1. Edge Types

The transformed graph contains edges. These edges are of the type gene-gene, disease-disease, and disease-gene. The main goal of this project is to predict disease-gene associations. Therefore, the primary edge type is disease-gene.

In this project, the machine learning model is trained to predict disease-gene associations. Gene-gene associations and disease-disease associations are auxiliary edges that assist in the training process of the machine learning model. Therefore, gene-gene associations and disease-disease associations are separated from disease-gene associations. This process is depicted in Figure 5.5.

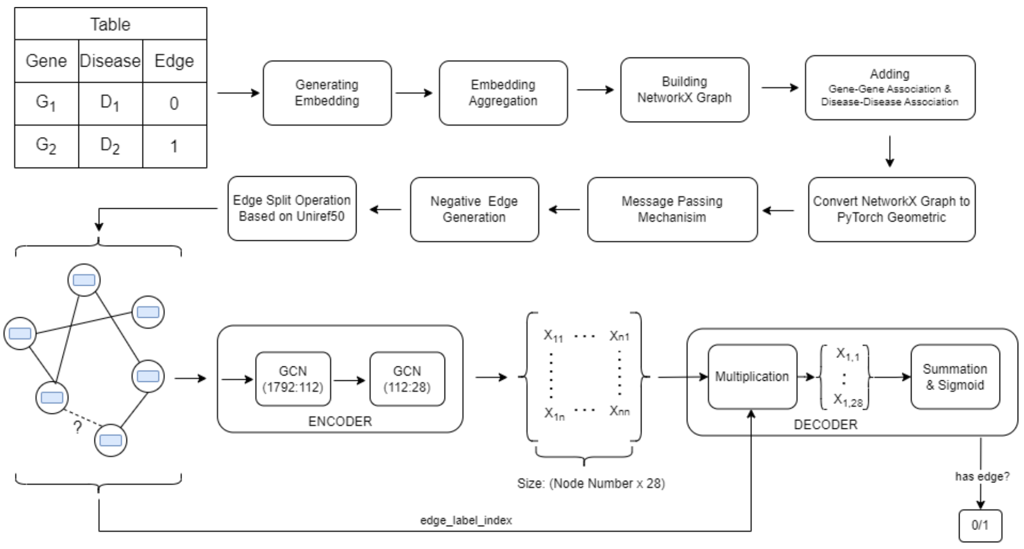


Figure 5.5 Main Algorithm of GLADIGATOR

5.5.2. Node Similarities

Proteins often exhibit similarities with other proteins. Likewise, diseases can exhibit similarities with other diseases. The partitioning process takes these similarities into account. In our project, partitioning operations are built on protein similarities.

The dataset partitioning operation is based on Uniref50[92]. Uniref50[92] clusters proteins based on protein sequences. There are also other clustered protein groups like Uniref100[92] and Uniref90[92]. These clusters differ from each other. The main difference is the percentage of sequence similarity. Uniref50[92] is built on 50% sequence similarity.

In the transformed graph, proteins and diseases are represented as nodes. Inside the nodes, ids, gene_symbol, and embeddings are stored. The graph is partitioned based on Uniref50 using the nodes' id and nodes' gene_symbol attributes.

5.5.3. Adding Negative Edges

In the link prediction model, both positive edges and negative edges are required to perform well. The original dataset contains positive edges, which means there is an edge between the start and stop nodes. The label of this edge is **1**. In negative edges, there is no edge between the start and stop nodes. This edge's label is **0**. Before the machine learning model training starts, adding negative edges to the original graph is a crucial step. This process is depicted in Figure 5.5.

5.5.4. Partition Ratio

The original graph is modified. Negative edges are constructed in the same number as positive edges. Negative edges are added to the original graph. After all steps are completed, the ratio of the parts is assigned. The training set contains edges that are 80% of the graph's edges. The validation set contains edges that are 10% of the graph's edges. The test set contains edges that are 10% of the graph's edges.

5.5.5. Dataset Split Operation

The operation of splitting the dataset is crucial for the performance of the machine learning model. There are various approaches to this operation. Generally, research separates the train, validation, and test sets, while others only separate the train and test sets.

In this research, the process of splitting into train, validation, and test sets is employed. The ratio of train, validation, and test split plays a key role in the performance of the machine learning model. A (8:1:1) ratio is used in this research. Detailed information is provided in Table 5.4.

Another significant factor in the performance of the machine learning model used to predict associations is the addition of negative edges to the graph. In this research, negative edges are constructed and added to the graph. The ratio for generating negative edges is (1:1), meaning the number of negative edges generated is equal to the number of edges in the graph. All positive and negative edges are between genes/proteins and diseases. Detailed information is provided in Table 5.4.

Graph ID	Train Positive Edge Number	Train Negative Edge Number	Validation Positive Edge Number	Validation Negative Edge Number	Test Positive Edge Number	Test Negative Edge Number
1	716	716	92	92	91	91
2	8108	8108	1024	1024	1020	1020
3	267858	267856	33841	33841	33561	33561
4	300190	300190	37872	37872	37636	37636
5	62130	62131	8096	8096	7872	7872
6	56754	56754	7170	7170	7140	7140

Table 5.4 Splitted Graph Detailed Information

5.6. Machine Learning Model

In this section, GLADIGATOR's machine learning model is explained detailly. There are two main subsection that are *Model Architecture* and *Model Hyperparameters Tuning*.

5.6.1. Model Architecture

Machine learning architecture refers to the structure and organization of components and processes. It encompasses how data is processed, how models are trained and evaluated, and how predictions are generated. One common type of model architecture is the encoder-decoder structure. In encoder-decoder structure, there are an encoder processes input data and extracts relevant features, while a decoder generates output based on those features. This architecture is often used in tasks like natural language processing, image generation, and sequence-to-sequence tasks. Specifically, proposed method GLADIGATOR's architecture incorporates this encoder-decoder structure, which allows it to handle complex tasks by transforming input data into a meaningful representation and then generating relevant output.

5.6.1.1. Encoder

The encoder part of the model utilizes a 2-layer Graph Convolutional Network (GCN) to transform the input node features into a latent representation.

First, we have the **Input Layer** denoted by $\mathbf{H}^{(0)}$. The input to the encoder is a feature matrix $\mathbf{X} \in \mathbb{R}^{N \times F}$, where N represents the number of nodes in the graph, and F represents the number of features for each node. Formally, we initialize the hidden representation as:

$$\mathbf{H}^{(0)} = \mathbf{X}$$

Next, the **First GCN Layer** processes these input features. This layer aggregates information from each node's neighbors to update the node representations. This is achieved using the adjacency matrix $\mathbf{A} \in \mathbb{R}^{N \times N}$, which encodes the graph structure. The transformation applied in this layer can be expressed as:

$$\mathbf{H}^{(1)} = \sigma\left(\mathbf{A}\mathbf{H}^{(0)}\mathbf{W}^{(0)}\right)$$

where $\mathbf{W}^{(0)}$ is the weight matrix for the first GCN layer, and σ is an activation function such as ReLU. This operation effectively mixes the features of neighboring nodes, allowing each node to gather information from its immediate neighborhood.

The **Second GCN Layer** further refines these node representations by repeating a similar process. It takes the output from the first GCN layer and again aggregates information based on the graph structure:

$$\mathbf{H}^{(2)} = \mathbf{A}\mathbf{H}^{(1)}\mathbf{W}^{(1)}$$

where $\mathbf{W}^{(1)}$ is the weight matrix for the second GCN layer. The output of this layer, $\mathbf{H}^{(2)}$, provides a more abstract representation of the nodes, capturing higher-order neighborhood information.

5.6.1.2. Decoder

The decoder part of the model is responsible for generating predictions about the relationships between pairs of nodes based on their latent representations from the encoder.

First, we perform **Element-wise Multiplication**. Let $\mathbf{h}_i$ and $\mathbf{h}_j$ be the node representations for nodes i and j from the encoder output $\mathbf{H}^{(2)}$. The element-wise multiplication of these two vectors is computed as:

$$\mathbf{e} = \mathbf{h}_i \odot \mathbf{h}_j$$

where $\odot$ denotes the element-wise multiplication. This operation produces a new vector $\mathbf{e}$ that combines the features of the two nodes in a pairwise manner, highlighting interactions between corresponding features of the nodes.

Next, we perform **Summation** of the resulting vector. The elements of $\mathbf{e}$ are summed to produce a single scalar value s:

$$s = \sum_{k=1}^{d} e_k$$

where e_k is the k-th element of $\mathbf{e}$ and d is the dimensionality of $\mathbf{H}^{(2)}$. This summation step reduces the vector to a scalar, aggregating the pairwise interactions into a single measure.

Finally, we apply a **Sigmoid Activation** function to this scalar value to obtain the final output $\mathbf{z}$:

$$\mathbf{z} = \sigma(s) = \frac{1}{1 + \exp(-s)}$$

The sigmoid function squashes the scalar s into a value between 0 and 1, which can be interpreted as a probability. This final output $\mathbf{z}$ represents the model's prediction about the relationship between nodes i and j (e.g., the likelihood of an edge existing between them).

5.6.2. Model Hyperparameters Tuning

The structure of the machine learning model is depicted in Figure 5.5. Apart from the structure, there are other crucial components of the machine learning model, which are discussed below.

Optimizer

The optimizer used in this project is the AdamW optimizer, with a learning rate of 0.001. A comparative analysis was conducted with other optimizers to determine the most effective one. The optimizer that yielded the best performance was **AdamW**. Other optimizers such as Adadelta, Adagrad, Adam [93], Adamax, ASGD [94], NAdam, RAdam, RMSprop, Rprop, and SGD [95] were evaluated [51]. These optimizers were tested with different learning rates: 0.1, 0.01, and 0.001. The results of this comparison are presented in Section 6.

AdamW

AdamW is an optimization algorithm that modifies the standard Adam algorithm by decoupling the weight decay from the gradient update. The standard Adam optimizer can be written as:

Parameter Update Rule:

$$\theta_t = \theta_{t-1} - \eta \cdot \frac{\hat{m}_t}{\sqrt{\hat{v}_t} + \epsilon}$$

where:

- η is the learning rate,
- $\hat{m}_t$ and $\hat{v}_t$ are bias-corrected first and second moment estimates:

$$\hat{m}_t = \frac{m_t}{1 - \beta_1^t} \quad \text{and} \quad \hat{v}_t = \frac{v_t}{1 - \beta_2^t}$$

- m_t and v_t are the first and second moment estimates:

$$m_t = \beta_1 \cdot m_{t-1} + (1 - \beta_1) \cdot g_t$$

$$v_t = \beta_2 \cdot v_{t-1} + (1 - \beta_2) \cdot g_t^2$$

- g_t is the gradient at time step t,

- β_1 and β_2 are the exponential decay rates for the moment estimates,
- ϵ is a small constant for numerical stability.

In AdamW, weight decay is applied directly to the weights before the update:

Modified Parameter Update Rule with Weight Decay:

$$\theta_t = \theta_{t-1} - \eta \cdot \left(\frac{\hat{m}_t}{\sqrt{\hat{v}_t} + \epsilon} + \lambda \cdot \theta_{t-1} \right)$$

where λ is the weight decay coefficient.

Loss Function

The loss function used in this project is BCEWithLogitsLoss[96]. Other loss functions were also evaluated to determine which one performs best. The loss function that yielded the best performance was **BCEWithLogitsLoss**. Loss functions such as L1Loss, MSELoss, PoissonNLLLoss, BCEWithLogitsLoss, HingeEmbeddingLoss, HuberLoss, SmoothL1Loss, and SoftMarginLoss were compared [97]. The results of this comparison are presented in Section 6.

BCEWithLogitsLoss

The BCEWithLogitsLoss combines a Sigmoid layer and the Binary Cross-Entropy (BCE) loss in a single class. This is numerically more stable than using a plain Sigmoid followed by a BCE loss. The loss function can be expressed as:

Binary Cross-Entropy Loss:

$$\text{BCE}(p, y) = -\big(y \cdot \log(p) + (1-y) \cdot \log(1-p)\big)$$

where:

- y is the ground truth label,
- p is the predicted probability.

With Logits: Instead of using the predicted probability p, BCEWithLogitsLoss takes the logits z (raw scores outputted by the model) and applies the Sigmoid function to them:

$$p = \sigma(z) = \frac{1}{1 + e^{-z}}$$

The combined loss function then becomes:

$$\text{BCEWithLogitsLoss}(z, y) = -\big(y \cdot \log(\sigma(z)) + (1 - y) \cdot \log(1 - \sigma(z))\big)$$

which simplifies to:

$$\text{BCEWithLogitsLoss}(z, y) = \max(z, 0) - z \cdot y + \log\left(1 + e^{-|z|}\right)$$

By combining the Sigmoid activation and BCE loss in one step, BCEWithLogitsLoss improves numerical stability and reduces the risk of overflow or underflow during computation.

6. EXPERIMENTAL RESULTS

This section elucidates the outcomes of the research. It encompasses information about the graph, the outcome of the machine learning model, and the performance evaluation.

6.1. Hyperparameter Optimization

Details about the customized graphs have been previously discussed. The structure of the machine learning model plays a crucial role in this research. This section elaborates on the machine learning model hyperparameters and configurations employed.

6.1.1. Hyperparamenters & Configuration

The hyperparameters significantly influences the research's performance. Various hyperparameters have been utilized in this research, as outlined in Table 6.1. All the networks listed in Table 6.1 employ a graph convolution network.

Network ID	Layer Number	Input Dense	Hidden Denses	Output Dense
1	2	1792	112	28
2	2	1792	224	14
3	3	1792	224-112	28
4	3	1792	224-56	8
5	4	1792	224-112-56	14
6	4	1792	224-112-28	8
7	5	1792	448-224-112-56	14
8	5	1792	448-224-56-14	8

Table 6.1 Network Structures for Hyperparameters Tuning

The network configuration plays a pivotal role in the performance of this research. It includes parameters such as optimizers and loss functions. This research utilizes several network optimizers. The network optimization is based on a comparison of optimizers and loss functions. The compared optimizers are Adadelta, Adagrad, Adam, AdamW, Adamax, ASGD, NAdam, RAdam, RMSprop, Rprop, SGD.

Conversely, the loss function is a critical parameter for network performance. Several loss functions have been compared. Compared loss functions are L1Loss, MSELoss, PoissonNLLLoss, BCEWithLogitsLoss, HingeEmbeddingLoss, HuberLoss, SmoothL1Loss, SoftMarginLoss.

6.1.2. Network Configuration Comparison

All network configurations have been tested. The decisive performance metrics are the best accuracy and the best F1 score. These scores are obtained from the test data results. The comparison results are presented in Table 6.2. The default loss function is used. The network id parameter is referenced from Table 6.1.

Optimizer ID	Graph ID	Network ID	Optimizer	Learning Rate	Loss Function	Accuracy	F1 Score
1	1	1	Adadelta	0.001	BCEWithLogitsLoss	0.538	0.670
2	1	1	Adagrad	0.001	BCEWithLogitsLoss	0.700	0.740
3	1	1	Adam	0.001	BCEWithLogitsLoss	0.716	0.754
4	**1**	**1**	**AdamW**	**0.001**	**BCEWithLogitsLoss**	**0.761**	**0.786**
5	1	1	Adamax	0.001	BCEWithLogitsLoss	0.722	0.745
6	1	1	ASGD	0.001	BCEWithLogitsLoss	0.672	0.746
7	1	1	NAdam	0.001	BCEWithLogitsLoss	0.722	0.742
8	1	1	RAdam	0.001	BCEWithLogitsLoss	0.683	0.720
9	1	1	RMSprop	0.001	BCEWithLogitsLoss	0.722	0.757
10	1	1	Rprop	0.001	BCEWithLogitsLoss	0.655	0.683
11	1	1	SGD	0.001	BCEWithLogitsLoss	0.560	0.670
12	1	1	Adadelta	0.01	BCEWithLogitsLoss	0.611	0.678
13	1	1	Adagrad	0.01	BCEWithLogitsLoss	0.672	0.720
14	1	1	Adam	0.01	BCEWithLogitsLoss	0.500	0.660
15	1	1	AdamW	0.01	BCEWithLogitsLoss	0.590	0.633
16	1	1	Adamax	0.01	BCEWithLogitsLoss	0.500	0.66
17	1	1	ASGD	0.01	BCEWithLogitsLoss	0.640	0.700
18	1	1	NAdam	0.01	BCEWithLogitsLoss	0.672	0.725
19	1	1	RAdam	0.01	BCEWithLogitsLoss	0.683	0.716
20	1	1	RMSprop	0.01	BCEWithLogitsLoss	0.600	0.065
21	1	1	Rprop	0.01	BCEWithLogitsLoss	0.711	0.717
22	1	1	SGD	0.01	BCEWithLogitsLoss	0.622	0.682
23	1	1	Adadelta	0.1	BCEWithLogitsLoss	0.640	0.690
24	1	1	Adagrad	0.1	BCEWithLogitsLoss	0.683	0.719
25	1	1	Adam	0.1	BCEWithLogitsLoss	0.500	0.666
26	1	1	AdamW	0.1	BCEWithLogitsLoss	0.460	0.600
27	1	1	Adamax	0.1	BCEWithLogitsLoss	0.600	0.650
28	1	1	ASGD	0.1	BCEWithLogitsLoss	0.622	0.688
29	1	1	NAdam	0.1	BCEWithLogitsLoss	0.677	0.707
30	1	1	RAdam	0.1	BCEWithLogitsLoss	0.616	0.672
31	1	1	RMSprop	0.1	BCEWithLogitsLoss	0.722	0.757
32	1	1	Rprop	0.1	BCEWithLogitsLoss	0.664	0.726
33	1	1	SGD	0.1	BCEWithLogitsLoss	0.650	0.715

Table 6.2 Comparison of the Optimizers

6.1.3. Loss Function Comparison

All loss functions have been tested. The decisive performance metrics are accuracy and F1 score. These scores are obtained from the test data results. The comparison results are presented in Table 6.3.

Loss Function ID	Graph ID	Network ID	Optimizer	Loss Function	Accuracy	F1-Score
1	1	1	AdamW (lr:0.001)	L1Loss	0.600	0.707
2	1	1	AdamW (lr:0.001)	MSELoss	0.550	0.670
3	1	1	AdamW (lr:0.001)	PoissonNLLLoss	0.600	0.485
4	**1**	**1**	**AdamW (lr:0.001)**	**BCEWithLogitsLoss**	**0.750**	**0.788**
5	1	1	AdamW (lr:0.001)	HingeEmbeddingLoss	0.550	0.441
6	1	1	AdamW (lr:0.001)	HuberLoss	0.566	0.690
7	1	1	AdamW (lr:0.001)	SmoothL1Loss	0.572	0.695
8	1	1	AdamW (lr:0.001)	SoftMarginLoss	0.500	0.660

Table 6.3 Comparison of the Loss Function

6.1.4. Hyperparameters Comparison

All hyperparameters have been tested. The decisive performance metrics are accuracy and F1 score. These scores are obtained from the test data results. The comparison results are presented in Table 6.4.

Network ID	Graph ID	Optimizer	Loss Function	Accuracy	F1-Score
1	**1**	**AdamW (lr:0.001)**	**BCEWithLogitsLoss**	**0.75**	**0.781**
2	1	AdamW (lr:0.001)	BCEWithLogitsLoss	0.727	0.758
3	1	AdamW (lr:0.001)	BCEWithLogitsLoss	0.722	0.747
4	1	AdamW (lr:0.001)	BCEWithLogitsLoss	0.710	0.752
5	1	AdamW (lr:0.001)	BCEWithLogitsLoss	0.70	0.740
6	1	AdamW (lr:0.001)	BCEWithLogitsLoss	0.711	0.761
7	1	AdamW (lr:0.001)	BCEWithLogitsLoss	0.688	0.743
8	1	AdamW (lr:0.001)	BCEWithLogitsLoss	0.722	0.764

Table 6.4 Comparison of the Network Arhitecture

6.2. Performance Evaluation of the Finalized Model

Within this subsection, the finalized model is subjected to a thorough performance evaluation. A series of established metrics and benchmarks are utilized to meticulously analyze the model's efficacy and precision. The evaluation is conducted in a manner that ensures a comprehensive understanding of the model's capabilities in simulating real-world scenarios.

6.2.1. Model Performance on Constructed Graphs

In this section, the performance of the machine learning model on GDA score based constructed graphs is elaborated. Prior to the explanation, the selected machine learning model's hyperparameters used to illustrate the performance of the machine learning model are mentioned below.

- **Machine Learning Model Architecture** : 2 layer graph convolutional network(1792:112 ; 112:28) is used by encoder-decoder.
- **Optimizer** : AdamW (learning rate: 0.001)
- **Loss Function** : BCEWithLogitsLoss

The performance of the machine learning model on GDA score based constructed graphs is evaluated. The validation results of the machine learning model are presented in Table 6.5. The test results of the machine learning model are presented in Table 6.6.

Graph ID	Accuracy	F1 Score	Precision	Recall	ROC-AUC	PR-AUC	Specificity
1	0.719	0.741	0.688	0.800	0.735	0.688	0.637
2	0.830	0.836	0.805	0.871	0.896	0.893	0.789
3	0.890	0.893	0.875	0.911	0.953	0.957	0.869
4	0.891	0.895	0.867	0.955	0.955	0.960	0.858

Table 6.5 Model Validation Performance Results on Graph 1-4

The performance of the machine learning model on various datasets is evaluated. The test results of the machine learning model are presented in Table 6.6.

Graph ID	Accuracy	F1 Score	Precision	Recall	ROC-AUC	PR-AUC	Specificity
1	0.761	0.786	0.711	0.877	0.807	0.746	0.644
2	0.827	0.840	0.784	0.900	0.900	0.900	0.750
3	0.890	0.892	0.874	0.911	0.953	0.957	0.869
4	0.888	0.892	0.867	0.918	0.954	0.959	0.860

Table 6.6 Model Independent Test Performance Results on Graph 1-4

6.2.2. Performance Comparison with the State-of-the-art

6.2.2.1. SkipGNN Comparison

It is essential to contrast this model with other research, particularly the SkipGNN: Predicting Molecular Interactions with Skip-Graph Networks[59] study.
In the SkipGNN study[59], the researchers developed a machine learning model to forecast drug-target interactions, drug-drug interactions, protein-protein interactions, and disease-gene interactions. Although various datasets are employed in this study, the disease-gene associations dataset is utilized for testing all the methods mentioned in 6.9.

In GLADIGATOR, the DisGeNet curated dataset[98] is used to evaluate the model's performance. In DisGeNet curated dataset, there are 19,783 nodes and 81,746 edges. Constructed graph contains 19,001 nodes and 78,097 edges. 3,649 associations are not formed in the customized dataset due to the inability to collect number of 782 disease and gene information from the proposed sources. Consequently, 3,649 associations could not be established in the customized dataset. These missing associations are decomposed into train, validation and test parts with the same splitting ratio (8:1:1).

The validation performance result is presented in Table 6.7, and the test performance result is in Table 6.8.

Graph ID	Accuracy	F1 Score	Precision	Recall	ROC-AUC	PR-AUC	Specificity
5	0.899	0.902	0.869	0.938	0.960	0.966	0.859

Table 6.7 Model Validation Performance Results on Graph 5

The machine learning model in this study is trained with same dataset with used in SkipGNN[59], which includes disease-gene associations information. Therefore, these two

Graph ID	Accuracy	F1 Score	Precision	Recall	ROC-AUC	PR-AUC	Specificity
5	0.889	0.893	0.859	0.930	0.955	0.960	0.848

Table 6.8 Model Independent Test Performance Results on Graph 5

studies are comparable based on disease-gene association prediction. The comparison results of the disease-gene association prediction methods are presented in Table 6.9.

Method Name	ROC-AUC	PR-AUC	References
GLADIGATOR	**0.950**	**0.956**	-
HOGCN	0.936	0.941	[60]
ResMGCN	0.925	0.935	[61]
MixHop	0.916	0.912	[66]
SkipGNN	0.912	0.915	[59]
struc2vec	0.909	0.910	[71]
GCN	0.906	0.909	[62]
SC	0.863	0.905	[68]
GIN	0.900	0.916	[64]
JK-Net	0.898	0.891	[65]
VGAE	0.873	0.902	[63]
node2vec	0.834	0.828	[70]
DeepWalk	0.832	0.827	[69]
L3	0.832	0.899	[67]

Table 6.9 Comparing Methods: Insights from the DisGeNet Curated Dataset

There are discrepancies between the performance results in Table 6.9 and Table 6.8 due to the different splitting ratios. In Table 6.9, the splitting ratio is (7:1:2), whereas in Table 6.8, the splitting ratio is (8:1:1).

6.2.2.2. Comparison on OGB

Protein-disease associations are collected from ogbl-biokg(OGB). The aim is to construct a new graph to execute the machine learning model on ogbl-biokg. The performance results of the machine learning model are presented with graph id 6 in Table 6.10 and Table 6.11.

Proposed method can be compared with other methods. However, other methods represent only MRR score. In our model, an encode-decode process is employed based on a two-layer graph convolutional network. Therefore, the machine learning model used in this study cannot reliably produce an MRR result. For each prediction, the output of decoder is one binary (0/1) value. Because of that, MRR calculation is not reliable for GLAGIGATOR.

Graph ID	**Accuracy**	**F1 Score**	**Precision**	**Recall**	**ROC-AUC**	**PR-AUC**	**Specificity**
6	0.859	0.867	0.819	0.920	0.941	0.945	0.797

Table 6.10 Model Validation Performance Results on Graph 6

Graph ID	**Accuracy**	**F1 Score**	**Precision**	**Recall**	**ROC-AUC**	**PR-AUC**	**Specificity**
6	0.874	0.883	0.822	0.955	0.963	0.966	0.793

Table 6.11 Model Independent Test Performance Results on Graph 6

GLADIGATOR was tested only on the protein-disease part of the ogbl-biokg(OGB) dataset. Other methods were tested covering the entire ogbl-biokg(OGB) dataset. The results are mentioned in Table 6.12.

Method	MRR Score	References
RelEns	**0.9618**	**[99]**
GLADIGATOR	0.859	-
ComplEx$\hat{2}$	0.8583	[100]
UniBi	0.8550	[101]
AutoBLM-KGBench	0.8536	[102]
ComplEx-RP	0.8492	[103]
TripleRE	0.8348	[104]
AutoSF	0.8309	[105]
PairRE	0.8164	[106]
ComplEx	0.8095	[107]
DistMult	0.8043	[108]
RotatE	0.7989	[109]
TransE	0.7452	[110]

Table 6.12 Comparing Methods: Insights from the OGB Datasets

6.3. Ablation Study

Ablation study is a terminology prevalent in fields like science and engineering, particularly in areas such as artificial intelligence and machine learning. Essentially, it is an analysis method involving various components, losses, or substitutions to discern a pattern. Adjustments and removals in the machine are carried out to identify the significant parts or components of the model. For instance, in a deep learning model, imputation might be conducted to supply specific layers or feature extractors. This technique is crucial for determining which quantities or components of the model have increased or expanded relative to what was anticipated. An ablation study serves as a vital instrument for identifying which features or parts should be concentrated on to enhance the overall performance or efficiency of the model. Detailed information about the ablation study is provided in Table 6.13.

Ablation Study Name	Description
Embedding Aggregation	Applying zero padding to different parts for gene/protein embeddings and disease embeddings Gene/Protein embeddings and disease embeddings are created from different universal sets.
Negative Edge Generation	Negative only disease-gene associations are generated. In the absence of these, negative edges are created between gene-gene, disease-disease and disease-gene nodes.

Table 6.13 Explanation of Ablation Study

6.3.1. Embedding Aggregation

In the scope of this thesis, Prot-T5[1] serves as the source for gene/protein embeddings, which are characterized by a dimensionality of (1 * 1024). In parallel, disease embeddings are derived from Biobert[2], exhibiting dimensions of (1 * 768). Standardization of these embeddings for machine learning utilization typically involves zero-padding, which, in this context, equates to appending 256 zeros to the terminus of the disease embeddings.

To facilitate the machine learning model's comprehension, a scaling operation is applied across all embeddings. This operation, while beneficial for model processing, may render the distinction between gene/protein and disease embeddings less discernible. Addressing this, the methodology introduced herein applies a differential zero-padding strategy to each embedding type. For gene/protein embeddings, zero-padding entails the addition of 768 zeros at the embedding's end. Conversely, for disease embeddings, zero-padding involves the prefixing of 1024 zeros. This approach ensures a uniform embedding size of (1 * 1792), thereby maintaining the network's structural integrity and enabling a more nuanced comparison of the zero-padding algorithm's efficacy.

The outcome of removing the embedding aggregation is documented in Table 6.14 and in Table 6.15. A performance comparison can be drawn by examining the second and third rows in Table 6.14 and in Table 6.15.

6.3.2. Negative Edge Generation

In the first implemented dataset, negative edges were created randomly and among all node types by PyTorch-Geometric[91]. Only the edges produced between the gene and the disease affected performance as a new feature. The generation of negative edges between only genes and diseases plays a significant role in the performance of the disease-gene association prediction. Creating negative disease-gene associations boosts the performance of the disease-gene association prediction model. A performance comparison can be made by looking at the first and second rows in Table 6.14 and Table 6.15.

Embedding Aggregation	Generating Negative Disease-Gene Edges	**Graph ID**	**Accuracy**	**F1 Score**	**Precision**	**Recall**	**ROC-AUC**
✓	✗	2	0.739	0.782	0.671	0.938	0.87
✗	✓	2	0.823	0.826	0.810	0.843	0.895
✓	✓	2	0.830	0.836	0.805	0.871	0.896

Table 6.14 Comparing Validation Performance in Ablation Studies

Embedding Aggregation	Generating Negative Disease-Gene Edges	**Graph ID**	**Accuracy**	**F1 Score**	**Precision**	**Recall**	**ROC-AUC**
✓	✗	2	0.75	0.8	0.675	0.974	0.925
✗	✓	2	0.823	0.835	0.783	0.893	0.916
✓	✓	2	0.828	0.84	0.784	0.904	0.904

Table 6.15 Comparing Test Performance in Ablation Studies

The results present the outcomes of the ablation studies, highlighting their significance for the methodology and their impact on model performance. The introduction of the Embedding Aggregation feature has simplified the differentiation of gene/protein embeddings from those of diseases, leading to enhanced method efficacy.

Furthermore, the exclusive creation of edges between genes and diseases has contributed to a boost in method performance. Upon examining the results, it becomes evident that the selective edge generation between genes and diseases contributes more significantly to performance enhancement than the embedding aggregation feature.

6.4. Use Case Study

In this section, we selected several positive and negative disease-gene pair predictions of GLADOGATOR and tried to verify them via literature validation, to assess the biological relevance of the proposed method. The model under consideration has been developed using Graph 1. A portion of the outcomes derived from this model is documented in Table 6.16. These associations' gene disease association score are equal or higher than 0.9 .

In Table 6.16, the initial six use cases were derived from the GLADIGATOR test outcomes, as depicted in Graph 1. These six instances were chosen at random from a total of 179 relationships. In addition, the final three use case details were sourced from existing literature. However, it is important to note that these three use cases are absent from the DisGeNet database. Consequently, GLADIGATOR underwent testing with data that is not included in the DisGeNet database.

It presents the data on the associations between specific genes and diseases in Table 6.16, together with the predictions produced by the GLADIATOR method and the validations from DisGeNet. Here's a detailed explanation of each column:

- **Gene Symbol**: The standard abbreviation for the gene in question.
- **Uniprot ID**: The unique identifier for the protein associated with the gene in the UniProt database.
- **Disease Name**: The name of the disease associated with the gene.
- **Disease CUI**: The Concept Unique Identifier (CUI) for the disease in the Unified Medical Language System (UMLS).
- **True Association Information**: Indicates whether there is verified information about the disease-gene association.

- **GLADIGATOR Prediction**: Indicates whether GLADIGATOR predicted an association between the gene and the disease.
- **DisGeNet Confirmation**: Indicates whether DisGeNet confirmed the association between the gene and the disease.

Gene Symbol	**Uniprot ID**	**Disease Name**	**Disease CUI**	**True Association Information**	**GLADIGATOR Prediction**	**DisGeNet Confirmation**
ABCA1	O95477	Tangier Disease	C0039292	✓	✓	✓
ABCA4	P78363	Stargardt's disease	C0271093	✓	✓	✓
ACADM	P11310	Medium-chain acyl-coenzyme A dehydrogenase deficiency	C0220710	✓	✓	✓
FANCA	O15360	Polycystic Kidney Disease I	C3149841	✗	✗	✗
PINK1	Q9BXM7	Herlitz Disease	C0079683	✗	✗	✗
NPR2	P20594	Polyneuropathy, Hearing Loss, Ataxia, Retinitis Pigmentosa, And Cataract	C2675204	✗	✗	✗
PADI4	Q9UM07	Rheumatoid Arthritis Disease	C0003873	✓	✓	✗
GATA2	P23769	Emberger Syndrome Disease	C1868560	✓	✓	✗
SETBP1	Q9Y6X0	Schinzel-Giedion Syndrome Disease	C3554436	✓	✓	✗

Table 6.16 Predictions from the Proposed Method Using Graph 1

6.4.1. ABCA1 Gene and Tangier Disease Association

The association between ABCA1 Gene and Tangier Disease was correctly predicted by GLADIGATOR. Tangier disease (TD) is a rare autosomal recessive disorder associated with genetic mutations in the ABCA1 gene [111]. This gene encodes a cell plasma membrane cholesterol transporter. Individuals with TD exhibit significantly reduced levels of plasma high-density lipoprotein cholesterol (HDL-C) and apolipoprotein A-1 (ApoA-I)[112]. Consequently, they face a moderately increased risk of cardiovascular disease due to their low HDL levels. Tangier disease (TD) typically leads to cholesterol accumulation in peripheral tissues and early coronary disease, although its clinical expression varies widely [113]. Scientific evidence for the association explained above are given below.

- Mutations in ABCA1 cause Tangier disease characterized by defective cholesterol homeostasis and high density lipoprotein (HDL) deficiency [114].
- ATP-binding cassette transporter A1 (ABCA1) gene mutations in a patient with Tangier disease, who presented an uncommon clinical history, and in his family were found and characterized [115].
- Homozygous variations in ATP-binding cassette transporter A1 typically cause Tangier disease, a rare autosomal recessive condition linked with several other abnormalities (eg, enlarged discolored tonsils) [116].

6.4.2. ABCA4 Gene and Stargardt's Disease Association

The association between ABCA4 Gene and Stargardt's Disease was correctly predicted by GLADIGATOR. Stargardt's disease (STGD) is an autosomal recessive retinopathy primarily caused by genetic mutations in the *ABCA4* gene located on chromosome 1 (OMIM 601691). The *ABCA4* gene encodes a retina-specific ATP-binding cassette transporter, which plays a crucial role in the processing of vitamin A and the maintenance of photoreceptor cells in the retina. Mutations in *ABCA4* result in impaired transport of vitamin A derivatives, leading to the accumulation of toxic lipofuscin in the retinal pigment epithelium (RPE) and subsequent photoreceptor degeneration [117]. Clinically, STGD is characterized by progressive central vision loss, macular atrophy, and the presence of yellowish flecks in the fundus. Notably, the mutation spectrum of *ABCA4* is diverse, with numerous missense, splicing, frameshift, and nonsense variants identified [118]. Recent studies, including comprehensive next-generation sequencing analyses, have revealed novel variants in the *ABCA4* gene, further expanding our understanding of the genetic basis of Stargardt's disease [119]. Scientific evidence for the association explained above are given below.

- Stargardt disease (STGD1, OMIM 248200) is a common hereditary juvenile or early adult onset macular degeneration [120].
- Mutations in the gene encoding ABCA4 are responsible for Stargardt disease (STGD1), an autosomal recessive retinal degenerative disease that causes severe vision loss [121].
- Null missense ABCR (ABCA4) mutations in a family with stargardt disease and retinitis pigmentosa [122].

6.4.3. ACADM Gene and Medium-chain Acyl-Coenzyme A Dehydrogenase Deficiency Disease Association

The association between ACADM Gene and Medium-chain Acyl-Coenzyme A Dehydrogenase Deficiency Disease was correctly predicted by GLADIGATOR. Medium-chain acyl-coenzyme A dehydrogenase deficiency (MCADD) is an autosomal recessive disorder caused by mutations in the ACADM gene. This gene encodes the medium-chain acyl-CoA dehydrogenase enzyme, which plays a crucial role in fatty acid metabolism. Individuals with MCADD have impaired ability to break down medium-chain fatty acids, leading to a buildup of toxic metabolites[123]. Clinically, MCADD presents with symptoms such as hypoglycemia, lethargy, vomiting, and seizures, often triggered by fasting or illness [124]. Newborn screening programs have been instrumental in early detection of MCADD, allowing for timely intervention through dietary modifications and avoiding fasting periods. Understanding the genetic basis of MCADD through the study of ACADM gene variants is essential for improving diagnosis, management, and genetic counseling for affected individuals [125]. Scientific evidence for the association explained above are given below.

- Medium-chain acyl-CoA dehydrogenase deficiency: Two novel ACADM mutations identified in a retrospective screening [126].
- Significance of ACADM mutations identified through newborn screening of MCAD deficiency in Japan [127].
- About 60% of MCADD patients are homozygous for the c.985A¿G (p.Lys329Glu) mutation in the ACADM gene (G985 allele) [128].

6.4.4. FANCA Gene and Polycystic Kidney Disease I Disease Association

GLADIGATOR correctly predicted that there is no association between the FANCA Gene and Polycystic Kidney Disease I Disease. The FANCA gene is known for its role in the Fanconi anemia (FA) pathway, which is crucial for DNA repair. Mutations in the FANCA gene can lead to Fanconi anemia, a condition characterized by bone marrow failure and increased risk of cancer [129, 130]. While the FANCA gene is not directly associated with Polycystic Kidney Disease I (PKD1), it's important to note that PKD1 is caused by mutations in the PKD1 gene, which encodes for polycystin-1, a protein involved in cell signaling and structure within the kidneys [131].

However, both conditions involve genetic mutations that affect cellular processes. In the case of PKD1, mutations lead to the formation of cysts in the kidneys, while in Fanconi anemia, mutations impair DNA repair mechanisms. It's also worth mentioning that individuals with Fanconi anemia may have a higher risk of developing various types of cancers, including those affecting the kidneys [129].

6.4.5. PINK1 Gene and Herlitz Disease Association

GLADIGATOR correctly predicted that there is no association between the PINK1 Gene and Herlitz Disease. The PINK1 gene is primarily associated with Parkinson's disease, particularly the early-onset form of the disorder which typically begins before age 50 [132]. Mutations in the PINK1 gene can alter or eliminate the kinase domain, leading to a loss of protein function, and at least one mutation affects the mitochondrial-targeting motif, potentially disrupting the delivery of the protein to mitochondria[132, 133].

Herlitz Disease, also known as Epidermolysis Bullosa Acquisita (EBA), is a rare chronic autoimmune blistering disease that affects the skin and mucous membranes. It is not directly associated with the PINK1 gene. Instead, Herlitz Disease is characterized by autoantibodies against type VII collagen, which is a component of anchoring fibrils that attach the epidermis to the dermis[134].

While both conditions involve genetic mutations and cellular dysfunction, they affect different systems and are not linked by the PINK1 gene.

6.4.6. NPR2 Gene and Polyneuropathy, Hearing Loss, Ataxia, Retinitis Pigmentosa, and Cataract Disease Association

GLADIGATOR correctly predicted that there is no association between the NPR2 Gene and Polyneuropathy, Hearing Loss, Ataxia, Retinitis Pigmentosa, and Cataract Disease. The NPR2 gene is not directly associated with the condition known as Polyneuropathy, Hearing Loss, Ataxia, Retinitis Pigmentosa, and Cataract (PHARC). PHARC is caused by homozygous or compound heterozygous mutations in the ABHD12 gene located on chromosome 20p11.21 [135]. This condition is a slowly progressive neurologic disorder that resembles Refsum disease and is characterized by a combination of symptoms including peripheral neuropathy, hearing loss, cataracts, and retinitis pigmentosa [135].

The NPR2 gene is involved in the regulation of iron metabolism and has been implicated in conditions related to iron overload, such as hemochromatosis [136]. It does not have a known association with the symptoms described in PHARC syndrome[137].

6.4.7. PADI4 Gene and Rheumatoid Arthritis Disease Association

GLADIGATOR correctly predicted that there is an association between the PADI4 Gene and Rheumatoid Arthritis Disease. The PADI4 gene is implicated in the development of Rheumatoid Arthritis (RA), a chronic autoimmune disease characterized by joint inflammation and destruction. PADI4 encodes an enzyme that modifies proteins by converting arginine residues into citrulline, a process known as citrullination [138]. This modification can lead to the production of neo-antigens that are targeted by the immune system in RA patients.

Research has identified specific polymorphisms and haplotypes within the PADI4 gene that are associated with an increased risk of RA. These genetic variations may influence the enzyme's activity, leading to higher levels of citrullinated proteins and, consequently, a more aggressive immune response [139]. The presence of anti-cyclic citrullinated peptide (anti-CCP) antibodies, which are linked to PADI4 activity, is a significant marker for RA and can be detected early in the disease's progression. Understanding the role of PADI4 in RA pathogenesis is crucial for developing targeted treatments and improving patient outcomes.

6.4.8. GATA2 Gene and Emberger Syndrome Disease Association

GLADIGATOR correctly predicted that there is an association between the GATA2 Gene and Emberger Syndrome Disease. The GATA2 gene is associated with Emberger Syndrome, a rare genetic disorder characterized by primary lymphedema and myelodysplastic syndromes/acute myeloid leukemia (MDS/AML). Emberger Syndrome is caused by inactivating mutations in one of the two parental GATA2 genes, leading to a reduction in the levels of the GATA2 transcription factor [140]. This haploinsufficiency results in a variety of clinical manifestations, including immunodeficiency, pulmonary disease, and vascular/lymphatic dysfunction [141].

Patients with Emberger Syndrome may present with severe viral infections, disseminated mycobacterial infections, and pulmonary alveolar proteinosis due to the deficiency of monocytes and other immune cells2. The syndrome also includes features such as lymphedema, which is a hallmark of the condition, and can progress to MDS/AML. Early genetic diagnosis is crucial for managing the disease and providing appropriate clinical care [142].

6.4.9. SETBP1 Gene and Schinzel-Giedion Syndrome Disease Association

GLADIGATOR correctly predicted that there is an association between the SETBP1 Gene and Schinzel-Giedion Syndrome Disease. The SETBP1 gene is associated with Schinzel-Giedion Syndrome (SGS), a rare genetic disorder characterized by profound neurodevelopmental delays, distinctive facial features, and multiple congenital malformations [143]. Mutations in the SETBP1 gene, particularly de novo mutations, lead to a loss-of-function of the gene, resulting in insufficient production of the SETBP1 protein. This protein plays a crucial role in brain development and function, and its deficiency is linked to the symptoms observed in SGS [144].

Individuals with SGS may exhibit a range of symptoms including absent speech, expressive language delays, intellectual disability, autistic traits or autism, developmental delays, ADHD, low muscle tone, seizures or EEG abnormalities, and various learning disabilities [145]. The condition is also associated with an increased risk of cancer [146]. Research into SETBP1 haploinsufficiency disorder continues to uncover more about the gene's function and the mechanisms behind the disorder's manifestation [144].

7. OPEN SOURCE MODEL

The method proposed is available in https://github.com/hubiodatalab/GLADIGATOR. This repository houses all the source codes. It contains three primary project folders: Main-Project, Comparison-SkipGNN, and Comparison-Open-Graph-Dataset. Detailed information is provided below, and all scripts are documented in markdown files.

7.1. Main-Project

Associations between genes and diseases, as well as associations between diseases, are obtained from DisGeNet. Information about gene-gene interactions is sourced from BioGrid. Detailed disease information is collected from UMLS, and gene embeddings are obtained from Prot-T5 from Uniprot. This project has three main folders: Gathering-Data, Build-Graph, and Run-Model.

- Gathering-Data: This folder houses the code for collecting dataset information from the internet. This part can be run with the call scripts that are mentioned below.

```
python3 gather_gene_disease_information.py
python3 gather_disease_data_from_umls.py
```

- Build-Graph: This folder contains the code used to construct the custom graph for the model training phase. This part can be run with the call scripts that are mentioned below.

```
python3 build_graph.py 0.5
```

- Run-Model: This folder contains the code used to train proposed model to predict gene-disease associations. This part can be run with the call scripts that are mentioned below.

```
python3 run_model.py "../../graph-files/Graph_Own_0.5.pt"
```

7.2. Comparison-SkipGNN

Gene-disease associations are obtained from a curated dataset from DisGeNet. Disease-disease associations are sourced from DisGeNet. Gene-gene information is collected from BioGrid, and detailed disease information is gathered from UMLS. Gene embeddings are obtained from Prot-T5 from Uniprot. This project has two main folders: Build-Graph and Run-Model.

- Build-Graph: This folder contains the code used to construct the custom graph for the model training phase. The source information is obtained from the SkipGNN repository. The curated dataset from DisGeNet is downloaded from the SkipGNN repository. This part can be run with the call scripts that are mentioned below.

```
python3 build_graph_skipgnn_comparison.py
```

- Run-Model: This folder contains the code used to train proposed model to predict gene-disease associations. This part can be run with the call scripts that are mentioned below.

```
python3 run_model_skipgnn_comparison.py
"../../graph-files/Graph_Comparison_SkipGNN.pt"
```

7.3. Comparison-Open-Graph-Dataset

Gene-disease associations are obtained from the ”ogbl-biokg” dataset from OGB. Disease-disease associations are sourced from DisGeNet. Gene-gene information is collected from BioGrid, and detailed disease information is gathered from UMLS. Gene embeddings are obtained from Prot-T5 from Uniprot. This project has two main folders: Build-Graph and Run-Model.

- Build-Graph: This folder contains the code used to construct the custom graph for the model training phase. The source information is obtained from the ”ogbl-biokg” dataset from OGB. Only the protein-disease part of this dataset is used. This part can be run with the call scripts that are mentioned below.

```
python3 build_graph_ogb_comparison.py
```

- Run-Model: This folder contains the code used to train proposed model to predict gene-disease associations. This part can be run with the call scripts that are mentioned below.

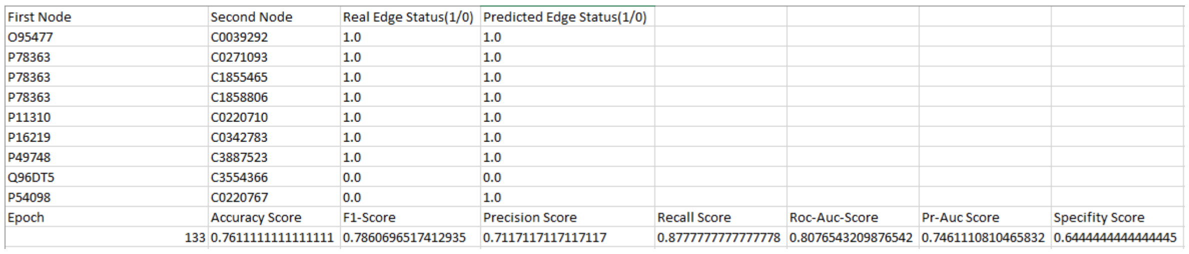
```
python3 run_model_ogb_comparison.py
"../../graph-files/Graph_Comparison_OGB.pt"
```

7.4. Output Format

Validation and test results are given as output in cvs format. Validation results are created with a name where the expressions ”val_results”, ”name of the graph used” are added one after the other. Test results are created with a name such as ”test_results”, ”name of the graph used” added one after the other.

End of the output that is in .csv format, contains 2 part. First part has 4 columns about prediction. First column contains first node id information. Second column contains second node id information. Third column contains real information about associations. Fourth column contains prediction information abput associations. The first part is place in beginning of the output.

The second part contains performance scores that are accuracy, f1-score, precision, recall, roc-auc, pr-auc and specifity. Also, second part contains that which epochs results are displayed. The second part is placed end of the first part. The validation results output format and the test results output format are used same format that is explained this section. Sample structure of the output format is mentioned in Figure 7.1.



First Node	Second Node	Real Edge Status(1/0)	Predicted Edge Status(1/0)				
O95477	C0039292	1.0	1.0				
P78363	C0271093	1.0	1.0				
P78363	C1855465	1.0	1.0				
P78363	C1858806	1.0	1.0				
P11310	C0220710	1.0	1.0				
P16219	C0342783	1.0	1.0				
P49748	C3887523	1.0	1.0				
Q96DT5	C3554366	0.0	0.0				
P54098	C0220767	0.0	1.0				
Epoch	Accuracy Score	F1-Score	Precision Score	Recall Score	Roc-Auc-Score	Pr-Auc Score	Specifity Score
133	0.7611111111111111	0.7860696517412935	0.7117117117117117	0.8777777777777778	0.8076543209876542	0.7461110810465832	0.6444444444444445

Figure 7.1 Sample Output Format

7.5. Generate Predictions

The GLADIGATOR models are stored in the Trained-Models directory. A script is available to facilitate predictions between genes and diseases. An example of how to use this script for predicting the relationship between a gene and a disease is provided. Executing the specified command will prompt the Graph_Own_0.9_model, a pre-trained GLADIGATOR model, to predict the association between the SETBP1 gene and the C3554436 disease.

```
python3 MakePrediction.py Graph_Own_0.9_model.pth
SETBP1 C3554436
```

8. DISCUSSION & CONCLUSION

This research is focused on creating a machine-learning model to predict the links between diseases and genes. In today's world, the widespread analysis of genetic data and health outcomes is crucial in advancing medical diagnosis and treatment. However, sustaining these advancements conventionally is often insufficient.

Machine learning holds substantial potential in this domain due to its capability to analyze vast volumes of data and discern intricate relationships. The significance of the model presented in this thesis lies in its ability to make more precise and sensitive predictions of potential associations.

GLADIGATOR extracts features from genetic datasets using pre-trained machine learning models. By utilizing embedding, graphs are constructed. Moreover, descriptions of diseases are instrumental in generating embedding to train the machine learning model with ease.

The foundational data underpinning these relationships, inclusive of disease-disease and disease-gene associations, were meticulously sourced from the DisGeNet database[98]. Concurrently, the gene-gene associations, with a focus on homo-sapiens, were curated from BioGrid [88].

The graph thus constructed was subsequently converted into a format amenable to machine learning models. This data served as the bedrock for training a model that boasts an encoder based on a 2-layer graph convolutional network, complemented by a decoder dedicated to training with labeled data. The validation and testing procedures mirrored the training regimen, ensuring consistency and reliability.

Various types of machine learning model structures were examined. The machine learning model has customized parameters that include the number of dense and encoder-decoder structures. Different types of optimizers and loss functions were tested. The best performance result was obtained from the 1st network id in Table 6.4. These performance tests were conducted using the Graph 1.

In addition, optimizers and loss functions were tested to select the best optimizer for the proposed methods. The AdamW optimizer performed best compared to other optimizers, as mentioned in Table 6.2. The BCEWithLogitsLoss loss function performed best compared to other loss functions, as mentioned in Table 6.3. These performance tests were conducted using the Graph 1.

The operation of splitting the graph dataset plays a key role in performance. The (8:1:1) ratio is a common split ratio. Therefore, the (8:1:1) ratio is applied for the proposed method. Moreover, adding negative edges to the input graph is a crucial factor in the performance of the model. Graph 1 is a smallest dataset. Graph 2, Graph 5, and Graph 6 are normal-sized graphs. Graph 3 and Graph 4 are large datasets.

To enhance the performance of machine learning models, it is imperative to utilize extensive datasets for training. This assertion is substantiated by the data presented in Tables 6.5 and Table 6.6. A comparative analysis of the model's performance reveals a suboptimal outcome when trained with Graph 1. In contrast, a notable improvement is observed when the model is trained with Graph 2. However, the model achieves its peak performance when the training involves Graph 3 and Graph 4. Both the validation and test results corroborate the superior performance of the GLADIGATOR when a substantial graph dataset is employed.

The GLADIGATOR model performance shows remarkable efficiency in Graph 1. However, when evaluating its performance on other graphs, it shows suboptimal results. This discrepancy can be attributed to two factors. First, the smaller size of Graph 1 may affect the generalization ability of the algorithm. Second, the presence of similar disease information in Graph 1 may hinder its performance during testing, especially when different diseases are encountered in the test data.

The Graph 1 has the lowest specificity at 0.644, indicating a relatively poor performance in correctly identifying negative cases. This could suggest issues such as data imbalance, improper feature selection, or model overfitting/underfitting. The Specificity improves progressively in the subsequent models, with Graph ID 3 achieving the highest Specificity at 0.869, closely followed by Graph ID 4 at 0.860. These higher values suggest that the latter models are significantly better at correctly classifying negative instances, likely due to better model tuning, feature selection, or handling of class imbalance. The improvement in specificity, alongside other metrics like ROC-AUC and PR-AUC, indicates a more robust and reliable model performance.

The GLADIGATOR's performance was evaluated on various datasets. The results indicate that the model's performance improved consistently across different graphs, with the highest accuracy, F1 score, precision, recall, ROC-AUC, PR-AUC, and specificity achieved on Graph 3 in both validation and test datasets.

The results garnered from the GLADIGATOR were exceptional. The GLADIGATOR has performed exceedingly well, leading to the following insights drawn from the constructed graph:

- The quantity of edges is a determinant of paramount importance in the performance of a machine learning model. The model's efficacy is seen to fluctuate with the number of edges. This observation emerged from a comparative analysis of gene disease scores at thresholds of 0.1 and 0.9, with the model comprising a higher edge count surpassing its counterpart with fewer edges.
- The quantity of edges is a pivotal factor in the dynamics of machine learning models. The model's performance showcased consistency irrespective of the edge count. This finding was deduced by juxtaposing gene disease scores at thresholds of 0.1 and 0.05.

The GLADIGATOR performs well with Graph 5, which is built on the curated dataset of DisGeNet. Validation results are mentioned in Table 6.7. Test results are mentioned in Table 6.8. In addition to this part, the split ratio of Graph 5 is (8:1:1). The original SkipGNN [59], split ratio is (7:1:2). Also, the performance of the machine learning model based on the (7:1:2) split ratio is mentioned in Table 6.9. The performance of the machine learning model with the (7:1:2) split ratio perform well, but the performance is lower than the (8:1:1) ratio of the machine learning model.

The GLADIGATOR perform best according to Table 6.9. Without HOGCN [60] and ResMGCN [61], other methods' performance results are gathered from SkipGNN [59]. Also, HOGCN [60] and ResMGCN [61], with the addition of a new approach to this area, perform better than SkipGNN [59]. The model's efficacy was appraised utilizing two metrics: Roc-AUC and Pr-AUC. Among the 14 disparate machine learning models that prognosticate disease-gene associations, the model delineated in this study emerged as the preeminent contender.

All methods are mentioned Table 6.9, use neighbourhood relationships. Also, the GLADIGATOR method uses neighbourhood relationships. But the main difference of the GLADIGATOR is that use node embedings to predict link between nodes. In the other side, SkipGNN[59] use neighbourhood relationships, calculate skip similarity to generate skip graph to train model. Skip similarity is a metric calculated on 2-degree neighborhood relationships [59]. Also, HOGCN[60], use metric like skip similarity metric that is k-degree neighborhood relationships. In addition, MixHop[66] can also use neighborhood relationships differently.

According to the test results in Table 6.12, GLADIGATOR achieved a good result. The other performance metrics are high, as mentioned in Table 6.10 and Table 6.11. GLADIGATOR is second best method based on MRR score that is mentioned in Table 6.12. Other methods are mentioned in Table 6.12, use all part of the ogbl-biokg dataset. The GLADIGATOR is used only protein-disase association part. Better results could be obtained if GLADIGATOR cover the entire dataset was performed.

The ablation study, as detailed in Table 6.13, provides a comprehensive understanding of the significant components of the model. Two key components were examined: the embedding aggregation and generating negative disease-gene associations.

The incorporation of embedding aggregation is pivotal to the methodology proposed. An empirical evaluation revealed a decline in performance metrics upon the exclusion of this feature. The impact of omitting embedding aggregation is systematically recorded in Tables

6.14 and 6.15. A comparative analysis of performance can be discerned by scrutinizing the second and third rows of these tables.

Furthermore, embedding aggregation plays a vital role in magnifying the distinction between gene and disease nodes. This is attributed to the fact that gene embeddings and disease embeddings are derived from disparate spaces. An augmentation in the disparity between gene and disease embeddings correlates with an enhancement in the performance of GLADIGATOR.

The generation of negative edges between only genes and diseases plays a significant role in the performance of the disease-gene association prediction. Creating negative only between gene and disease nodes boosts the performance of the disease-gene association prediction model. A performance comparison can be made by looking at the first and third rows in Table 6.14 and in Table 6.15.

The model under consideration was developed using Graph 1, as detailed in Table 5.2. A portion of the outcomes derived from this model are documented in Table 6.16. Explanations for actual cases were provided in this section. These instances are referenced in Table 6.16.

The table provided includes disease-gene associations sourced from two different methodologies: predictions gathered from GLADIGATOR and results from a trained model. The first three rows list associations predicted by GLADIGATOR, a tool used for predicting disease-gene associations. These predictions include associations between the genes ABCA1, ABCA4, and ACADM with Tangier Disease, Stargardt's disease, and Medium-chain acyl-coenzyme A dehydrogenase deficiency, respectively. These associations are marked with a check (✓) indicating an established link. The subsequent six rows present associations determined by a trained model, which include genes such as FANCA, PINK1, NPR2, PADI4, GATA2, and SETBP1 with various diseases like Polycystic Kidney Disease I, Herlitz Disease, Polyneuropathy, Hearing Loss, Ataxia, Retinitis Pigmentosa, Cataract, Rheumatoid Arthritis Disease, Emberger Syndrome Disease, and Schinzel-Giedion Syndrome Disease. This portion of the table also distinguishes between established associations (✓) and those without a confirmed link (✗). This comprehensive table serves multiple use cases, including enhancing the accuracy of genetic research, aiding clinical diagnosis, supporting drug development, and integrating data for bioinformatics applications. The inclusion of UniProt IDs and Disease CUIs adds an extra layer of specificity, facilitating the use of these associations in various biomedical research and healthcare applications.

A detailed discussion of the system's limitations is provided in the subsequent section.

The limitations of GLADIGATOR are multifaceted. The efficacy of reliable predictions hinges on the quality and comprehensiveness of the utilized dataset. In scenarios where the dataset is incomplete or biased, the likelihood of inaccurate predictions escalates. Furthermore, the intricate nature of biological systems, coupled with the complexity of disease-gene associations, presents a formidable challenge that GLADIGATOR may not fully encapsulate. The process of feature selection is pivotal; however, incorrect choices in this regard can precipitate overfitting, characterized by the model's commendable performance on training data but subpar results on unseen data. Another concern is the generalizability of predictions. When GLADIGATOR is trained on specific datasets, its applicability may not extend well to other datasets or populations. Additionally, the risk of selection bias looms if the dataset does not adequately represent the broader population or if disease status is misclassified. Lastly, the requirement for significant computational resources by advanced machine learning models may not be within reach for all researchers, potentially curtailing the reproducibility and scalability of research. Addressing these challenges is imperative to bolster the reliability and utility of machine learning-based predictions in the realm of biomedical research.

To enhance the system's performance in future research, several improvements could be considered:

Refinement of the *embedding generator algorithms* is essential for more precise and representative feature extraction, potentially involving advanced machine learning techniques or optimization of existing methods. Improving the *encoder and decoder processes* by re-evaluating and possibly overhauling the current mechanisms could lead to increased precision and efficiency, possibly through the use of more sophisticated neural network models or hyperparameter tuning. Investigating alternative *network architectures* may reveal more effective data processing and learning methods, such as experimenting with different neural network types like convolutional or recurrent architectures. *Graph construction modifications*, including reassessment and potential redesign of the source data for the graph, could result in a more robust and contextually relevant graph. *Updating data sources* by adopting the latest versions of databases like DisGeNet and BioGrid ensures access to the most up-to-date and comprehensive data, with regular updates providing new insights and strengthening the system's knowledge base. The system's adaptability and relevance are further enhanced by the flexibility to change data sources. Lastly, *expanding data source* coverage to include genetic information from other species would widen the system's applicability, allowing for comparative studies and a deeper understanding of genetic patterns across various organisms. These enhancements collectively aim to bolster the system's analytical capabilities and overall performance.

REFERENCES

[1] Christian Dallago, Konstantin Schütze, Michael Heinzinger, Tobias Olenyi, Maria Littmann, Amy X. Lu, Kevin K. Yang, Seonwoo Min, Sungroh Yoon, James T. Morton, and Burkhard Rost. Learned embeddings from deep learning to visualize and predict protein sets. *Current Protocols*, 1(5):e113, **2021**. doi:https://doi.org/10.1002/cpz1.113.

[2] Jinhyuk Lee, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang. Biobert: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics*, 36(4):1234–1240, **2020**.

[3] Disgenet database info. `https://www.disgenet.org/dbinfo` [Accessed: (27/02/2024)].

[4] Image sources of chromosome, dna, gene. `https://en.wikipedia.org/wiki/Gene` [Accessed: (27/02/2024)].

[5] Image sources of radioactive fluorescent seq. `https://en.wikipedia.org/wiki/DNA_sequencing` [Accessed: (27/02/2024)].

[6] Making protein-process. `https://medlineplus.gov/genetics/understanding/howgeneswork/makingprotein/` [Accessed: (27/02/2024)].

[7] Image sources of scanning electron micrograph of mycobacterium tuberculosis. `https://en.wikipedia.org/wiki/Disease` [Accessed: (27/02/2024)].

[8] Simon Tripp and Martin Grueber. The economic impact and functional applications of human genetics and genomics. **2021**. doi:10.13140/RG.2.2.26362.41921.

[9] Graphs in data structure and algorithm. https://www.boardinfinity.com/blog/graphs-in-data-structure/ [Accessed: (27/02/2024)].

[10] Junxi Feng, Xiaohai He, Qizhi Teng, Chao Ren, Honggang Chen, and Yang Li. Reconstruction of porous media from extremely limited information using conditional generative adversarial networks. *Physical Review E*, 100, **2019**. doi:10.1103/PhysRevE.100.033308.

[11] Victor Zhou. Machine learning for beginners: An introduction to neural networks. *Towards Data Science*, 12, **2019**.

[12] Muhammad Usman Hadi, Rizwan Qureshi, Ayesha Ahmed, and Nadeem Iftikhar. A lightweight corona-net for covid-19 detection in x-ray images. *Expert Systems with Applications*, 225:120023, **2023**. doi:10.1016/j.eswa.2023.120023.

[13] Graph convolutional network. https://tkipf.github.io/graph-convolutional-networks/ [Accessed: (27/02/2024)].

[14] Roc curve. https://developers.google.com/machine-learning/crash-course/classification/roc-and-auc [Accessed: (27/02/2024)].

[15] Roc-auc & pr-auc metrics. https://neptune.ai/blog/f1-score-accuracy-roc-auc-pr-auc [Accessed: (27/02/2024)].

[16] McKinsey. The next wave of healthcare innovation, **2020**.

[17] Joseph Loscalzo, Isaac Kohane, and Albert-Laszlo Barabasi. Human disease classification in the postgenomic era: a complex systems approach to human pathobiology. *Molecular systems biology*, 3(1):124, **2007**.

[18] NCI. Types of cancer treatment, **2024**.

[19] Nature. Genes and disease, **2024**.

[20] National Human Genome Research Institute. Genetic disorders, **2024**.

[21] Cystic Fibrosis Foundation. Types of cftr mutations, **2023**.

[22] Mayo Clinic. Genetic testing, **2024**.

[23] National Human Genome Research Institute. Genetic disease research branch, **2022**.

[24] Ming He, Chen Huang, Bo Liu, Yadong Wang, and Junyi Li. Factor graph-aggregated heterogeneous network embedding for disease-gene association prediction. *BMC Bioinformatics*, 22(1):165, **2021**.

[25] Kenneth Opap and Nicola Mulder. Recent advances in predicting gene–disease associations. *F1000Research*, 6, **2017**.

[26] Karola Stotz and Paul Griffiths. Genes: philosophical analyses put to the test. *Hist. Philos. Life Sci.*, 26(1):5–28, **2004**.

[27] Manolis Kellis, Barbara Wold, Michael P Snyder, Bradley E Bernstein, Anshul Kundaje, Georgi K Marinov, Lucas D Ward, Ewan Birney, Gregory E Crawford, Job Dekker, Ian Dunham, Laura L Elnitski, Peggy J Farnham, Elise A Feingold, Mark Gerstein, Morgan C Giddings, David M Gilbert, Thomas R Gingeras, Eric D Green, Roderic Guigo, Tim Hubbard, Jim Kent, Jason D Lieb, Richard M Myers, Michael J Pazin, Bing Ren, John A Stamatoyannopoulos, Zhiping Weng, Kevin P White, and Ross C Hardison. Defining functional DNA elements in the human genome. *Proc. Natl. Acad. Sci. U. S. A.*, 111(17):6131–6138, **2014**.

[28] Sam Behjati and Patrick S Tarpey. What is next generation sequencing? *Arch. Dis. Child. Educ. Pract. Ed.*, 98(6):236–238, **2013**.

[29] Adam R Abate, Tony Hung, Ralph A Sperling, Pascaline Mary, Assaf Rotem, Jeremy J Agresti, Michael A Weiner, and David A Weitz. DNA sequence analysis with droplet-based microfluidics. *Lab Chip*, 13(24):4864–4869, **2013**.

[30] Yu-Chieh Wang, Suzanne E Peterson, and Jeanne F Loring. Protein post-translational modifications and regulation of pluripotency in human stem cells. *Cell Res.*, 24(2):143–160, **2014**.

[31] Gert C Scheper, Marjo S Van Der Knaap, and Christopher G Proud. Translation matters: protein synthesis defects in inherited disease. *Nature Reviews Genetics*, 8(9):711–723, **2007**.

[32] The causes of disease. `https://www.britannica.com/science/human-disease/The-causes-of-disease` [Accessed: (27/02/2024)].

[33] Iwao Milton Moriyama, Dean E Krueger, and Jeremiah Stamler. *Cardiovascular diseases in the United States*, volume 10. Harvard University Press, **1971**.

[34] Ammar Al-Chalabi and Christopher CJ Miller. Neurofilaments and neurological disease. *Bioessays*, 25(4):346–355, **2003**.

[35] Nicola Meola, Vincenzo Alessandro Gennarino, and Sandro Banfi. micrornas and genetic diseases. *Pathogenetics*, 2:1–14, **2009**.

[36] Melissa J Armstrong and Michael S Okun. Diagnosis and treatment of parkinson disease: a review. *Jama*, 323(6):548–560, **2020**.

[37] Annette Pruss-Ustun, Carlos F Corvalán, World Health Organization, et al. *Preventing disease through healthy environments: towards an estimate of the environmental burden of disease*. World Health Organization, **2006**.

[38] Matteo Marcantonio, Emily L Pascoe, and Frédéric Baldacchino. Sometimes scientists get the flu. wrong…! *Trends in parasitology*, 33(1):7–9, **2017**.

[39] Jörg Menche, Amitabh Sharma, Maksim Kitsak, Susan Dina Ghiassian, Marc Vidal, Joseph Loscalzo, and Albert-László Barabási. Uncovering disease-disease relationships through the incomplete interactome. *Science*, 347(6224):1257601, **2015**.

[40] Pamela B Davis. Cystic fibrosis since 1938. *American journal of respiratory and critical care medicine*, 173(5):475–482, **2006**.

[41] Alan R Cohen, Renzo Galanello, Dudley J Pennell, Melody J Cunningham, and Elliott Vichinsky. Thalassemia. *ASH Education Program Book*, 2004(1):14–34, **2004**.

[42] Ebony B Bookman, Kimberly McAllister, Elizabeth Gillanders, Kay Wanke, David Balshaw, Joni Rutter, Jill Reedy, Daniel Shaughnessy, Tanya Agurs-Collins, Dina Paltoo, Audie Atienza, Laura Bierut, Peter Kraft, M Daniele Fallin, Frederica Perera, Eric Turkheimer, Jason Boardman, Mary L Marazita, Stephen M Rappaport, Eric Boerwinkle, Stephen J Suomi, Neil E Caporaso, Irva Hertz-Picciotto, Kristen C Jacobson, William L Lowe, Lynn R Goldman, Priya Duggal, Megan R Gunnar, Teri A Manolio, Eric D Green, Deborah H Olster, Linda S Birnbaum, and for the NIH G $\times$ E Interplay Workshop participants. Gene-environment interplay in common complex diseases: forging an integrative model—recommendations from an NIH workshop. *Genet. Epidemiol.*, 35(4):217–225, **2011**.

[43] Allen D Roses. Pharmacogenetics and the practice of medicine. *Nature*, 405(6788):857–865, **2000**.

[44] Margaret E Hunter, Sean M Hoban, Michael W Bruford, Gernot Segelbacher, and Louis Bernatchez. Next-generation conservation genetics and biodiversity monitoring. *Evol. Appl.*, 11(7):1029–1034, **2018**.

[45] Eugene H. Kaji and Jeffrey M. Leiden. Gene and Stem Cell Therapies. *JAMA*, 285(5):545–550, **2001**. ISSN 0098-7484. doi:10.1001/jama.285.5.545.

[46] Helen K Brittain, Richard Scott, and Ellen Thomas. The rise of the genome and personalised medicine. *Clin. Med.*, 17(6):545–551, **2017**.

[47] Douglas Brent West et al. *Introduction to graph theory*, volume 2. Prentice hall Upper Saddle River, **2001**.

[48] P Zhang and G Chartrand. Introduction to graph theory. *Tata McGraw-Hill*, 2:2–1, **2006**.

[49] Batta Mahesh. Machine learning algorithms-a review. *International Journal of Science and Research (IJSR).[Internet]*, 9(1):381–386, **2020**.

[50] Prajit Ramachandran, Barret Zoph, and Quoc V Le. Searching for activation functions. *arXiv preprint arXiv:1710.05941*, **2017**.

[51] Optimizers. `https://pytorch.org/docs/stable/optim.html#:~:text=torch.Tensor%20s.-,Algorithms,-Adadelta` [Accessed: (27/02/2024)].

[52] Zewen Li, Fan Liu, Wenjie Yang, Shouheng Peng, and Jun Zhou. A survey of convolutional neural networks: analysis, applications, and prospects. *IEEE transactions on neural networks and learning systems*, 33(12):6999–7019, **2021**.

[53] Jiuxiang Gu, Zhenhua Wang, Jason Kuen, Lianyang Ma, Amir Shahroudy, Bing Shuai, Ting Liu, Xingxing Wang, Gang Wang, Jianfei Cai, et al. Recent advances in convolutional neural networks. *Pattern recognition*, 77:354–377, **2018**.

[54] Keiron O'shea and Ryan Nash. An introduction to convolutional neural networks. *arXiv preprint arXiv:1511.08458*, **2015**.

[55] Si Zhang, Hanghang Tong, Jiejun Xu, and Ross Maciejewski. Graph convolutional networks: a comprehensive review. *Computational Social Networks*, 6(1):1–23, **2019**.

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

[57] Felix Wu, Amauri Souza, Tianyi Zhang, Christopher Fifty, Tao Yu, and Kilian Weinberger. Simplifying graph convolutional networks. In Kamalika Chaudhuri

and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 6861–6871. PMLR, **2019**.

[58] Dragomir R Radev, Hong Qi, Harris Wu, and Weiguo Fan. Evaluating web-based question answering systems. In *LREC*. Citeseer, **2002**.

[59] Kexin Huang, Cao Xiao, Lucas Glass, Marinka Zitnik, and J. Sun. Skipgnn: predicting molecular interactions with skip-graph networks. *Scientific Reports*, 10, **2020**. doi:10.1038/s41598-020-77766-9.

[60] K. KC, R. Li, F. Cui, and A. R. Haake. Predicting biomedical interactions with higher-order graph convolutional networks. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 19(02):676–687, **2022**. ISSN 1557-9964. doi:10.1109/TCBB.2021.3059415.

[61] Zecheng Yin. Resmgcn: Residual message graph convolution network for fast biomedical interactions discovering. *arXiv preprint arXiv:2311.07632*, **2023**.

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

[63] Thomas N Kipf and Max Welling. Variational graph auto-encoders. *arXiv preprint arXiv:1611.07308*, **2016**.

[64] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? *arXiv preprint arXiv:1810.00826*, **2018**.

[65] Keyulu Xu, Chengtao Li, Yonglong Tian, Tomohiro Sonobe, Ken-ichi Kawarabayashi, and Stefanie Jegelka. Representation learning on graphs with jumping knowledge networks. In *International conference on machine learning*, pages 5453–5462. PMLR, **2018**.

[66] Sami Abu-El-Haija, Bryan Perozzi, Amol Kapoor, Nazanin Alipourfard, Kristina Lerman, Hrayr Harutyunyan, Greg Ver Steeg, and Aram Galstyan.

Mixhop: Higher-order graph convolutional architectures via sparsified neighborhood mixing. In *international conference on machine learning*, pages 21–29. PMLR, **2019**.

[67] István A Kovács, Katja Luck, Kerstin Spirohn, Yang Wang, Carl Pollis, Sadie Schlabach, Wenting Bian, Dae-Kyum Kim, Nishka Kishore, Tong Hao, et al. Network-based prediction of protein interactions. *Nature communications*, 10(1):1240, **2019**.

[68] Lei Tang and Huan Liu. Leveraging social media networks for classification. *Data Mining and Knowledge Discovery*, 23:447–478, **2011**.

[69] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 701–710. **2014**.

[70] Aditya Grover and Jure Leskovec. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 855–864. **2016**.

[71] Leonardo FR Ribeiro, Pedro HP Saverese, and Daniel R Figueiredo. struc2vec: Learning node representations from structural identity. In *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 385–394. **2017**.

[72] Rolf Apweiler, Amos Bairoch, Cathy H Wu, Winona C Barker, Brigitte Boeckmann, Serenella Ferro, Elisabeth Gasteiger, Hongzhan Huang, Rodrigo Lopez, Michele Magrane, et al. Uniprot: the universal protein knowledgebase. *Nucleic acids research*, 32(suppl_1):D115–D119, **2004**.

[73] Heidi L Rehm, Jonathan S Berg, Lisa D Brooks, Carlos D Bustamante, James P Evans, Melissa J Landrum, David H Ledbetter, Donna R Maglott, Christa Lese Martin, Robert L Nussbaum, et al. Clingen—the clinical genome resource. *New England Journal of Medicine*, 372(23):2235–2242, **2015**.

[74] Allan Peter Davis, Cynthia J Grondin, Robin J Johnson, Daniela Sciaky, Roy McMorran, Jolene Wiegers, Thomas C Wiegers, and Carolyn J Mattingly. The comparative toxicogenomics database: update 2019. *Nucleic acids research*, 47(D1):D948–D954, **2019**.

[75] David Tamborero, Carlota Rubio-Perez, Jordi Deu-Pons, Michael P Schroeder, Ana Vivancos, Ana Rovira, Ignasi Tusquets, Joan Albanell, Jordi Rodon, Josep Tabernero, et al. Cancer genome interpreter annotates the biological and clinical relevance of tumor alterations. *Genome medicine*, 10:1–8, **2018**.

[76] Alba Gutiérrez-Sacristán, Solene Grosdidier, Olga Valverde, Marta Torrens, Alex Bravo, Janet Pinero, Ferran Sanz, and Laura I Furlong. Psygenet: a knowledge platform on psychiatric disorders and their genes. *Bioinformatics*, 31(18):3075–3077, **2015**.

[77] Stanley JF Laulederkind, G Thomas Hayman, Shur-Jen Wang, Jennifer R Smith, Victoria Petri, Matthew J Hoffman, Jeff De Pons, Marek A Tutaj, Omid Ghiasvand, Monika Tutaj, et al. A primer for the rat genome database (rgd). *Eukaryotic Genomic Databases: Methods and Protocols*, pages 163–209, **2018**.

[78] Cynthia L Smith, Judith A Blake, James A Kadin, Joel E Richardson, Carol J Bult, and Mouse Genome Database Group. Mouse genome database (mgd)-2018: knowledgebase for the laboratory mouse. *Nucleic acids research*, 46(D1):D836–D842, **2018**.

[79] Sebastian Köhler, Leigh Carmody, Nicole Vasilevsky, Julius O B Jacobsen, Daniel Danis, Jean-Philippe Gourdine, Michael Gargano, Nomi L Harris, Nicolas Matentzoglu, Julie A McMurry, et al. Expansion of the human phenotype ontology (hpo) knowledge base and resources. *Nucleic acids research*, 47(D1):D1018–D1027, **2019**.

[80] Melissa J Landrum and Brandi L Kattman. Clinvar at five years: Delivering on the promise. *Human mutation*, 39(11):1623–1630, **2018**.

[81] Jacqueline MacArthur, Emily Bowler, Maria Cerezo, Laurent Gil, Peggy Hall, Emma Hastings, Heather Junkins, Aoife McMahon, Annalisa Milano, Joannella Morales, et al. The new nhgri-ebi catalog of published genome-wide association studies (gwas catalog). *Nucleic acids research*, 45(D1):D896–D901, **2017**.

[82] Mulin Jun Li, Zipeng Liu, Panwen Wang, Maria P Wong, Matthew R Nelson, Jean-Pierre A Kocher, Meredith Yeager, Pak Chung Sham, Stephen J Chanock, Zhengyuan Xia, et al. Gwasdb v2: an update database for human genetic variants identified by genome-wide association studies. *Nucleic acids research*, 44(D1):D869–D876, **2016**.

[83] Markus Bundschus, Mathaeus Dejori, Martin Stetter, Volker Tresp, and Hans-Peter Kriegel. Extraction of semantic biomedical relations from text using conditional random fields. *BMC bioinformatics*, 9(1):1–14, **2008**.

[84] Kwang-Il Goh, Michael E Cusick, David Valle, Barton Childs, Marc Vidal, and Albert-László Barabási. The human disease network. *Proceedings of the National Academy of Sciences*, 104(21):8685–8690, **2007**.

[85] A Bravo, M Cases, N Queralt-Rosinach, F Sanz, LI Furlong, et al. A knowledge-driven approach to extract disease-related biomarkers from the literature. *BioMed research international*, 2014, **2014**.

[86] Àlex Bravo, Janet Piñero, Núria Queralt-Rosinach, Michael Rautschka, and Laura I Furlong. Extraction of relations between genes and diseases from text and large-scale data analysis: implications for translational research. *BMC bioinformatics*, 16:1–17, **2015**.

[87] Rajaraman Anand and Ullman Jeffrey David. *Mining of massive datasets*. Cambridge university press, **2011**.

[88] Rose Oughtred, Chris Stark, Bobby-Joe Breitkreutz, Jennifer Rust, Lorrie Boucher, Christie Chang, Nadine Kolas, Lara O'Donnell, Genie Leung,

Rochelle McAdam, et al. The biogrid interaction database: 2019 update. *Nucleic acids research*, 47(D1):D529–D541, **2019**.

[89] Weihua Hu, Matthias Fey, Marinka Zitnik, Yuxiao Dong, Hongyu Ren, Bowen Liu, Michele Catasta, and Jure Leskovec. Open graph benchmark: Datasets for machine learning on graphs, **2021**.

[90] Aric Hagberg, Pieter Swart, and Daniel S Chult. Exploring network structure, dynamics, and function using networkx. Technical report, Los Alamos National Lab.(LANL), Los Alamos, NM (United States), **2008**.

[91] Matthias Fey and Jan E. Lenssen. Fast graph representation learning with PyTorch Geometric. In *ICLR Workshop on Representation Learning on Graphs and Manifolds*. **2019**.

[92] Baris E Suzek, Yuqi Wang, Hongzhan Huang, Peter B McGarvey, Cathy H Wu, and UniProt Consortium. Uniref clusters: a comprehensive and scalable alternative for improving sequence similarity searches. *Bioinformatics*, 31(6):926–932, **2015**.

[93] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, **2014**.

[94] Jiajin Li and Baoxiang Wang. Policy optimization with second-order advantage information. *arXiv preprint arXiv:1805.03586*, **2018**.

[95] Sebastian Ruder. An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747*, **2016**.

[96] Bcewithlogitsloss. `https://pytorch.org/docs/stable/generated/torch.nn.BCEWithLogitsLoss.html#bcewithlogitsloss:~:text=Shortcuts-,BCEWITHLOGITSLOSS,-CLASS` [Accessed: (27/02/2024)].

[97] Loss functions. https://pytorch.org/docs/stable/nn.html#loss-functions:~:text=of%20input%20matrices.-,Loss%20Functions,-nn.L1Loss [Accessed: (27/02/2024)].

[98] Janet Piñero, Juan Manuel Ramírez-Anguita, Josep Saüch-Pitarch, Francesco Ronzano, Emilio Centeno, Ferran Sanz, and Laura I Furlong. The disgenet knowledge platform for disease genomics: 2019 update. *Nucleic acids research*, 48(D1):D845–D855, **2020**.

[99] Ling Yue, Yongqi Zhang, Quanming Yao, Yong Li, Xian Wu, Ziheng Zhang, Zhenxi Lin, and Yefeng Zheng. Relation-aware ensemble learning for knowledge graph embedding. *arXiv preprint arXiv:2310.08917*, **2023**.

[100] Lorenzo Loconte, Nicola Di Mauro, Robert Peharz, and Antonio Vergari. How to turn your knowledge graph embeddings into generative models. *Advances in Neural Information Processing Systems*, 36, **2024**.

[101] Jiayi Li, Ruilin Luo, Jiaqi Sun, Jing Xiao, and Yujiu Yang. Prior bilinear based models for knowledge graph completion. *arXiv preprint arXiv:2309.13834*, **2023**.

[102] Yongqi Zhang, Quanming Yao, and James T Kwok. Bilinear scoring function search for knowledge graph learning. *IEEE transactions on pattern analysis and machine intelligence*, 45(2):1458–1473, **2022**.

[103] Yihong Chen, Pasquale Minervini, Sebastian Riedel, and Pontus Stenetorp. Relation prediction as an auxiliary training objective for improving multi-relational graph representations. *arXiv preprint arXiv:2110.02834*, **2021**.

[104] Long Yu, Zhicong Luo, Huanyong Liu, Deng Lin, Hongzhu Li, and Yafeng Deng. Triplere: Knowledge graph embeddings via tripled relation vectors. *arXiv preprint arXiv:2209.08271*, **2022**.

[105] Yongqi Zhang, Quanming Yao, Wenyuan Dai, and Lei Chen. Autosf: Searching scoring functions for knowledge graph embedding. In *2020 IEEE 36th International Conference on Data Engineering (ICDE)*, pages 433–444. IEEE, **2020**.

[106] L Chao, J He, T Wang, and W Chu. Pairre: Knowledge graph embeddings via paired relation vectors. arxiv 2020. *arXiv preprint arXiv:2011.03798*, **2020**.

[107] Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. Complex embeddings for simple link prediction. In *International conference on machine learning*, pages 2071–2080. PMLR, **2016**.

[108] Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. Embedding entities and relations for learning and inference in knowledge bases. *arXiv preprint arXiv:1412.6575*, **2014**.

[109] Zhiqing Sun, Zhi-Hong Deng, Jian-Yun Nie, and Jian Tang. Rotate: Knowledge graph embedding by relational rotation in complex space. *arXiv preprint arXiv:1902.10197*, **2019**.

[110] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. Translating embeddings for modeling multi-relational data. *Advances in neural information processing systems*, 26, **2013**.

[111] Sofía Barbosa-Gouveia, Silvia Fernández-Crespo, Héctor Lazaré-Iglesias, Arturo González-Quintela, Néstor Vázquez-Agra, and Álvaro Hermida-Ameijeiras. Association of a novel homozygous variant in abca1 gene with tangier disease. *Journal of Clinical Medicine*, 12(7), **2023**. ISSN 2077-0383. doi:10.3390/jcm12072596.

[112] Tangier disease (td). `https://www.ncbi.nlm.nih.gov/medgen/C0039292` [Accessed: (27/02/2024)].

[113] Tangier disease (td). `https://rarediseases.info.nih.gov/diseases/7731/tangier-disease` [Accessed: (27/02/2024)].

[114] Faraz Quazi and Robert S Molday. Differential phospholipid substrates and directional transport by ATP-binding cassette proteins ABCA1, ABCA7, and ABCA4 and disease-causing mutants. *J. Biol. Chem.*, 288(48):34414–34426, **2013**.

[115] Marianna Maranghi, Gessica Truglio, Antonio Gallo, Elvira Grieco, Antonella Verrienti, Anna Montali, Pietro Gallo, Francesco Alesini, Marcello Arca, and Marco Lucarelli. A novel splicing mutation in the abca1 gene, causing tangier disease and familial hdl deficiency in a large family. *Biochemical and biophysical research communications*, 508(2):487–493, **2019**.

[116] Yaser Carcora, Robert D Brook, Linda Farhat, Cristen J Willer, Melvyn Rubenfire, and Daniel Seung Kim. A novel homozygous abca1 variant in an asymptomatic man with profound hypoalphalipoproteinemia. *Journal of Clinical Lipidology*, 12(4):878–882, **2018**.

[117] Fang-Yuan Hu, Jian-kang Li, Feng-Juan Gao, Yu-He Qi, Ping Xu, Yong-Jin Zhang, Dan-Dan Wang, Lu-Sheng Wang, Wei Li, Sheng-Hai Zhang, et al. Abca4 gene screening in a chinese cohort with stargardt disease: identification of 37 novel variants. *Frontiers in Genetics*, 10:468295, **2019**.

[118] Stargardt disease (stgd). `https://www.aao.org/education/disease-review/stargardt-disease-stgd` [Accessed: (27/02/2024)].

[119] Stargardt disease (stgd). `https://www.ncbi.nlm.nih.gov/medgen/C0271093` [Accessed: (27/02/2024)].

[120] Qin Xiang, Yanna Cao, Hongbo Xu, Yi Guo, Zhijian Yang, Lu Xu, Lamei Yuan, and Hao Deng. Identification of novel pathogenic abca4 variants in a han chinese family with stargardt disease. *Bioscience Reports*, 39(1):BSR20180872, **2019**.

[121] Laurie L Molday, Daniel Wahl, Marinko V Sarunic, and Robert S Molday. Localization and functional characterization of the p.Asn965Ser (N965S)

ABCA4 variant in mice reveal pathogenic mechanisms underlying stargardt macular degeneration. *Hum. Mol. Genet.*, 27(2):295–306, **2018**.

[122] N F Shroyer, R A Lewis, A N Yatsenko, and J R Lupski. Null missense ABCR (ABCA4) mutations in a family with stargardt disease and retinitis pigmentosa. *Invest. Ophthalmol. Vis. Sci.*, 42(12):2757–2761, **2001**.

[123] Medium-chain acyl-coenzyme a dehydrogenase deficiency. `https://rarediseases.org/gard-rare-disease/medium-chain-acyl-coenzyme-a-dehydrogenase-deficiency/` [Accessed: (27/02/2024)].

[124] Medium-chain acyl-coenzyme a dehydrogenase deficiency. `https://www.ncbi.nlm.nih.gov/medgen/C0220710` [Accessed: (27/02/2024)].

[125] Acadm gene - acyl-coa dehydrogenase medium chain. `https://www.genecards.org/cgi-bin/carddisp.pl?gene=ACADM` [Accessed: (27/02/2024)].

[126] Andraz Smon, Urh Groselj, Marusa Debeljak, Mojca Zerjav Tansek, Sara Bertok, Magdalena Avbelj Stefanija, Katarina Trebusak Podkrajsek, Tadej Battelino, and Barbka Repic Lampret. Medium-chain acyl-CoA dehydrogenase deficiency: Two novel ACADM mutations identified in a retrospective screening. *J. Int. Med. Res.*, 46(4):1339–1348, **2018**.

[127] Keiichi Hara, Go Tajima, Satoshi Okada, Miyuki Tsumura, Reiko Kagawa, Kenichiro Shirao, Yoshinori Ohno, Shin'ichiro Yasunaga, Motoaki Ohtsubo, Ikue Hata, Nobuo Sakura, Yosuke Shigematsu, Yoshihiro Takihara, and Masao Kobayashi. Significance of ACADM mutations identified through newborn screening of MCAD deficiency in japan. *Mol. Genet. Metab.*, 118(1):9–14, **2016**.

[128] F V Ventura, P Leandro, A Luz, I A Rivera, M F B Silva, R Ramos, H Rocha, A Lopes, H Fonseca, A Gaspar, L Diogo, E Martins, E Leão-Teles, L Vilarinho,

and I Tavares de Almeida. Retrospective study of the medium-chain acyl-CoA dehydrogenase deficiency in portugal. *Clin. Genet.*, 85(6):555–561, **2014**.

[129] Jasmine D Peake and Eishi Noguchi. Fanconi anemia: current insights regarding epidemiology, cancer, and dna repair. *Human Genetics*, 141(12):1811–1836, **2022**.

[130] Fanca gene. `https://medlineplus.gov/genetics/gene/fanca/` [Accessed: (27/02/2024)].

[131] Genetic conditions and genes. `https://www.natera.com/wp-content/uploads/2021/07/Renasight-Conditions-List-2.pdf` [Accessed: (27/02/2024)].

[132] Pink1 gene. `https://medlineplus.gov/download/genetics/gene/pink1.pdf` [Accessed: (27/02/2024)].

[133] Suchita Ganesan and Venkatachalam Deepa Parvathi. Deconstructing the molecular genetics behind the pink1/parkin axis in parkinson's disease using drosophila melanogaster as a model organism. *Egyptian Journal of Medical Human Genetics*, 22:1–20, **2021**.

[134] Benjamin O'Callaghan, John Hardy, and Helene Plun-Favreau. Pink1: From parkinson's disease to mitophagy and back again. *PLoS Biology*, 21(6):e3002196, **2023**.

[135] Polyneuropathy, hearing loss, ataxia, retinitis pigmentosa, and cataract. `https://www.omim.org/entry/612674` [Accessed: (27/02/2024)].

[136] Npr2 gene. `https://www.uniprot.org/uniprotkb/P20594/entry` [Accessed: (27/02/2024)].

[137] Polyneuropathy-hearing loss-ataxia-retinitis pigmentosa-cataract syndrome. `https://www.orpha.net/en/disease/detail/171848` [Accessed: (27/02/2024)].

[138] Padi4 gene. https://www.genecards.org/cgi-bin/carddisp.pl?gene=PADI4 [Accessed: (27/02/2024)].

[139] Rheumatoid arthritis disease. https://www.mayoclinic.org/diseases-conditions/rheumatoid-arthritis/symptoms-causes/syc-20353648 [Accessed: (27/02/2024)].

[140] Gata2 gene. https://ashpublications.org/blood/article/123/6/809/32599/GATA2-deficiency-a-protean-disorder-of [Accessed: (27/02/2024)].

[141] Gata2 gene. https://ashpublications.org/blood/article/141/13/1524/493407/The-spectrum-of-GATA2-deficiency-syndrome [Accessed: (27/02/2024)].

[142] Gata2 gene. https://www.cincinnatichildrens.org/service/g/genetics-genomics-diagnostic-lab/molecular-genetics/custom-gene-sequencing/gata2 [Accessed: (27/02/2024)].

[143] Emanuela Leonardi, Elisa Bettella, Maria Federica Pelizza, Maria Cristina Aspromonte, Roberta Polli, Clementina Boniver, Stefano Sartori, Donatella Milani, and Alessandra Murgia. Identification of setbp1 mutations by gene panel sequencing in individuals with intellectual disability or with "developmental and epileptic encephalopathy". *Frontiers in Neurology*, 11:593446, **2020**.

[144] Setbp1 gene. https://www.setbp1.org/setbp1/ [Accessed: (27/02/2024)].

[145] Jordan H Whitlock, Tabea M Soelter, Timothy C Howton, Elizabeth J Wilk, Vishal H Oza, and Brittany N Lasseigne. Cell-type-specific gene expression and regulation in the cerebral cortex and kidney of atypical setbp1 s858r

schinzel giedion syndrome mice. *Journal of Cellular and Molecular Medicine*, 27(22):3565–3577, **2023**.

[146] Rocio Acuna-Hidalgo, Pelagia Deriziotis, Marloes Steehouwer, Christian Gilissen, Sarah A Graham, Sipko Van Dam, Julie Hoover-Fong, Aida B Telegrafi, Anne Destree, Robert Smigiel, et al. Overlapping setbp1 gain-of-function mutations in schinzel-giedion syndrome and hematologic malignancies. *PLoS genetics*, 13(3):e1006683, **2017**.