الجمهورية الجزائرية الديمقراطية الشعبية

PEOPLE'S DEMOCRATIC REPUBLIC OF ALGERIA

وزارة البحث العالمي والبحث العلمي

MINISTRY OF HIGHER EDUCATION AND SCIENTIFIC RESEARCH

University Yahia farés Médéa جامعة يحي فارس المدية

faculty of sciences كلية العلوم

قسم الرياضيات و الإعلام الآلي

DEPARTMENT OF MATHEMATICS AND COMPUTER SCIENCE

Specialty: Systems Engineering and Web Technology

Thesis for obtaining the Master's degree

THEME

Some Mathematical Perspective of Graph Neural Network

Proposed and Directed

Dr. BENTERKI Abdessalem.

Dr. HANTABLI Hamza.

Prepared and Presented by:

* LACHBOUR Keltoum.
* SLAMA Zineb.

Academic Year: 2023-2024

Acknowledgment

First, I would like to thank ALLAH S.W.T. for all the achievements that I have gained today.

Next, I wish to extend my grateful appreciation to all those who have contributed directly and indirectly to the preparation of this study.

I would like to take this opportunity to thank my supervisor, Dr. Benterki Abdessalem and Dr. Hentali Hamza.

DEDICATION

Last but not a least, I Wanna Thank Me, I Thank Me for believing in me , I Wanna Thank Me for don't give up , I Wanna Thank Me for getting back up after failure, I Wanna Thank me for never quitting , I Wanna Thank me for motivate me when I fail ,I Thank Me for trying to do this ,I Wanna Thank me for bieng giver and trying to give more I receive, I Wanna Thank Me for trying to do right than wrong , I Wanna Thank Me for just being me all the time .

I dedicated this modest work:

To my dear mother.

To my dear and my sweety BOUCHRA.

For my brother Amar.

To my sisters fatma, saidia, khaira, fatiha, zohra.

To my grandmother.

For all my family members without exception.

To my dear little Abdessalam and chifaa.

To LiLa, Ridha, Ayoub.

To All my friends.

To everyone who knows me.

By the way, we would like to say that we are lucky to have people who love us.

KeLTouM

Dedication

I dedicate this humble work:

To my mother may God have mercy on her soul

To my father may God prolong his life

To my brothers and dear teachers each by name

To everyone who supports me from near or far.

Zineb Slama.

Abstract

Graph Neural Networks (GNNs) have become highly effective tools for molecular property prediction, which is essential for drug research, material science, and environmental science applications. The Researchers use GNN to increase the precision of molecular property predictions through the representation of molecules as graphs, in which bonds are edges and atoms are nodes, for GNNs are able to capture the intricate interactions seen in molecular structures. The Combining between deep learning and cheminformatics improves Quantitative Structure-Activity Relationships (QSAR) and other prediction models. The goal of this work is to present a novel approach to molecular property prediction with GNNs, which will overcome the drawbacks of conventional computational methods and lessen the necessity for lengthy experimental testing.

Key words:

Graph Neural Network, GNN, molecular property prediction, Deep learning, Molecular representation, Graph, Quantitative Structure-Activity Relationship, QSAR.

Résumé

Les réseaux de neurones graphiques (GNN) sont devenus des outils très efficaces pour la prédiction des propriétés moléculaires, essentielles à la recherche sur les médicaments, à la science des matériaux et aux applications des sciences de l'environnement. Les chercheurs utilisent GNN pour augmenter la précision des prédictions des propriétés moléculaires grâce à la représentation des molécules sous forme de graphiques, dans lesquels les liaisons sont des bords et les atomes sont des nœuds, car les GNN sont capables de capturer les interactions complexes observées dans les structures moléculaires. La combinaison entre l'apprentissage profond et la chemininformatique améliore les relations quantitatives structure-activité (QSAR) et d'autres modèles de prédiction. Le but de ce travail est de présenter une nouvelle approche de prédiction des propriétés moléculaires avec les GNN, qui surmontera les inconvénients des méthodes informatiques conventionnelles et réduira la nécessité de longs tests expérimentaux.

Mots Clés:

Les réseaux de neurones graphiques, GNN, prédiction des propriétés moléculaires, Apprentissage profond, Graphe, Quantitative Structure-Activity Relationship, QSAR.

ملخص

أصبحت الشبكات العصبية البيانية (GNNs) أدوات فعالة للغاية للتنبؤ بالخصائص الجزيئية، وهو أمر ضروري لأبحاث الأدوية وعلوم المواد وتطبيقات العلوم البيئية. يستخدم الباحثون GNN لزيادة دقة تنبؤات الخصائص الجزيئية من خلال تمثيل الجزيئات كرسوم بيانية، حيث تكون الروابط عبارة عن حواف والذرات عبارة عن عقد، لأن شبكات GNN قادرة على التقاط التفاعلات المعقدة التي تظهر في الهياكل الجزيئية. يؤدي الجمع بين التعلم العميق والمعلوماتية الكيميائية إلى تحسين العلاقات الكمية بين الهيكل والنشاط (QSAR) ونماذج التنبؤ الأخرى. الهدف من هذا العمل هو تقديم نهج جديد للتنبؤ بالخصائص الجزيئية باستخدام شبكات GNN، والذي سيتغلب على عيوب الطرق الحسابية التقليدية ويقلل من ضرورة إجراء اختبارات تجريبية مطولة.

الكلمات الدالة

الشبكة العصبية للرسم البياني،GNN ، التنبؤ بالخصائص الجزيئية، التعلم العميق، التمثيل الجزيئي، الرسم البياني، العلاقة الكمية بين البنية والنشاط، QSAR.

Title	page
ACKNOWLEDGMENTS	I
DEDICATION	II
ABSTRACT	IV
LIST OF TABLES	VII
LIST OF FIGURES	X
TABLE OF SYMBOLS	XI
LIST OF ABBREVIATIONS	XI
LIST OF TERMINOLOGIES	XII

General Introduction.

Chapter 1

1 Introduction: 1
2 Overview 2
3 Background of the problem 2
4 Problem Statement 2
5 Objectives of the Research: 3
6 Scope of the Research 3
7 Theses Out line 3
8 Summary 4

Chapter 2

1 Introduction 5
2 Basic of the graph 7
2.1 Whatis the graph 7

2.2 Graph components: 7
2.2.1 The node: 7
2.2.2 The edge: 7
2.3 Types of Graphs: 7
2.4 Graph representations: 10
3 Deep Learning 12
3.1 Artificial Intelligence (AI): 12
3.1.1 Definition: 12
3.1.2 Types of AI: 13
3.2 Machine Learning (ML): 13
3.2.1 Definition: 13
3.2.2 Type: 13
3.3 Deep Learning (DL) 15
3.3.1 Definition: 15
3.3.2 Application of DL: 16
3.4 Differences between AI, machine learning and deep learning: 16
3.5 Convolutional neural network (CNN): 18
3.5.1 Definition: 18
3.5.2 CNN Architecture: 18
3.5.3 Advantages of Convolutional Neural Networks (CNNs): 19
3.5.4 Disadvantages of Convolutional Neural Networks (CNNs): 19
4 Graph neural networks 19
4.1 Graph neural networks definition: 19
4.2 The Type of Graph Neural Networks Architecture: 20
4.2.1 Graph Convolutional Network (GCN): 20
4.2.2 Graph Attention Networks (GAT): 21
4.2.3 GraphSAGE: 23
4.2.4 Graph Recurrent Neural Networks (GRNNs): 23
4.2.5 Equation of GNN Models: 24
4.3 Application of Graph Neural Networks: 24
4.4 How GNN works? 25
4.5 Where we can use GNN? 26
5 Definition of Chemoinformatics: 29
6 Molecular Representations: 30

7 Molecular similarity measurement: 31
7.1 Introduction: 31
7.2 Definition: 32
7.3 Similarity Coefficients: 32
7.4 Fingerprint similarity: 34
8 Quantitative structure- activity relationships: 35
8.1 Definition: 35
8.2 The Steps of QSAR study: 35
8.3 Advantages and disadvantages of QSAR: 36
8.3.1 Advantages: 36
8.3.2 Disadvantages: 36
9 Molecular property prediction: 36
9.1 Definition: 36
9.2 Where we can use MPP: 36
9.3 Challenges Molecular Property Prediction: 37
10 Graph neural networks for molecular property prediction: 37
11 Conclusion: 39

Chapter 3

1 Introduction: 40
2 Development environment: 40
2.1 Python: 40
2.2 Python module used: 41
2.3 Google Colaboratory: 43
2.3.1 Google Colaboratory definition: 44
2.4 Google Drive: 45
3 Summary: 46

Chapter 4

1 Introduction: 47
2 Explanation of our approach: 47
2.1 Dataset: 47
2.2 Data download: 47

3 Data preprocessing: 49
3.1 Splitting the data into stratified train and test sets: 49
4 Develop the Model..... 49
4.1 GCN model: 49
4.2 Evaluation:..... 51
5 Result model 53
5.1 The Result of Property 53
5.2 Table of Result model: 58
5.3 Comparison between the established model and the piper model: 60
5.3.1 THE Observation..... 60
5.3.2 The conclusion 61
6 Summary 61

Conclusion General

Conclusion general 62

Reference

Reference 63

List of figures:

Figure 1: Graph components. 7
Figure 2: Directed and undirected graphs. 8
Figure 3: Type of Graph..... 9
Figure 4: Representation of Undirected Graph to Adjacency Matrix 10
Figure 5: Representation of Directed Graph to Adjacency Matrix. 11
Figure 6: Machine Learning..... 13
Figure 7: Types of Machine leaning. 15
Figure 8: Application of DL..... 16
Figure 9: Represents The Relation Between AI, ML and Deep Learning. 18
Figure 10: A Simple Architecture of CNN. 18
Figure 11: A graph neural network architecture design. 20
Figure 12: working GCN. 21
Figure 13: Working of GAT..... 22
Figure 14: Architecture of Recurrent Graph Neural Network. 23
Figure 15: The Equations of GNN Models. 24
Figure 16: Difference between graph and node classification. 25
Figure 17: GNN Application. 27
Figure 18: Different Types of Molecular Representations..... 30

Figure 19: Level of Molecular Representations. 31
Figure 20: A molecular fingerprint representation 34
Figure 21: Graph neural networks for molecular property prediction 38
Figure 22: Google Drive applications and functions. 46
Figure 23: The dataset used. 47
Figure 24: Dataset example from the Dataset. 48
Figure 25: Plot of Deep Learning Model. 51
Figure 26 : The error rate of the model. 52
Figure 27 : Train logKOA 52
Figure 28 : Test LogKOA 53
Figure 29:Train LogVP 52
Figure 30: Test LogVP 53
Figure 31: Train LogMoler 53
Figure 32: Test LogMoler 54
Figure 33:train logOH 53
Figure 34:Test logOH..... 54
Figure 35: Train LogBCF..... 54
Figure 36: Test LogBCF 55
Figure 37: Train BP. 54
Figure 38: Test BP. 55
Figure 39: Train LogHL 55
Figure 40: Test LogHL. 56
Figure 41: train LogKmHL 55
Figure 42: test LogKmHL 56
Figure 43: Train MP 56
Figure 44: Test MP 57
Figure 45 : Train LogKOC 56
Figure 46:Test Log KOC 57
Figure 47: Train LogHalfLife 57
Figure 48: Test LogHalfLife 58

List of tables:

Table 1: differences between Artificial Intelligence, Machine Learning and Deep Learning: 16
Table 2: Application of Graph Neural Network: 27
Table 3: The most commonly used similarity or distance coefficients for similarity searching in chemical database. 32
Table 4: The Tanimoto Similarity Measures 34
Table 5: The number of training and testing dataset for each property. 49
Table 6 : The structure of the model. 50
Table 7: Result the Established model. 58
Table 8: The result of piper model 60

List of Acronyms:

1D: One Dimensional.

2D: Two Dimensional.

3D: Three Dimensional.

AI: Artificial Intelligence.

CNN: Convolutional Neural Network.

DL: Deep Learning.

GAT: Graph Attention Networks

GCNN: Graph Convolutional Neural Networks. .

GNN: Graph Neural Network.

GRNN: Graph Recurrent Neural Networks.

ML: Machine Learning.

MPP: Molecular property prediction.

QSAR: Quantitative structure-activity relationship.

SMILES: Simplified Molecular Input Line Entry System.

LIST OF TERMINOLOGIES:

Chemical Database: Chemical database is a database specifically designed to store chemical compound attributed data.

Molecule: The simplest structural unit of a substance that retains the properties of the substance, and is composed of one or more atoms.

Molecular Descriptors: Vector of number based on predefined attributes. It is used to represent a chemical structure.

Similarity Coefficient: Measure used to quantify the degree of structural resemblance between target structure and each structure in the database.

Atom: An atom is the smallest unit of ordinary matter that forms a chemical element. Every solid, liquid, gas, and plasma is composed of neutral or ionized atoms. Atoms are extremely small.

General Introduction

General Introduction

Through the convergence of cheminformatics, deep learning, and graph theory in recent years, new paths for scientific inquiry and application have become possible. Graphs, mathematical structures consisting of vertices (nodes) and edges, are used in many areas to model complex interactions. Where it is considered Graphs are essential for data representation in computer science, biology, and social networks Because of their structural adaptability.

Deep learning is a branch of machine learning that deals with multi-layered neural networks that are capable of autonomous learning and intelligent decision-making. Deep learning models can reach great levels of accuracy, frequently outperforming typical machine learning techniques, by utilizing massive volumes of data and sophisticated algorithms.

Such graph-structured data can be processed by a sophisticated class of deep learning models called graph neural networks (GNNs). These networks achieve remarkable performance in the tasks like node classification, link prediction, and graph classification by exploiting the intricate links found in the data. GNNs have proven their ability to handle complicated relational data structures by finding applications in a wide range of domains, including as social network analysis, healthcare, network security, and recommender systems.

In parallel, cheminformatics—an interdisciplinary area that combines computer science, information science, and chemistry—has been big developing. For turn chemical data into information that can be put to use, in material science and drug development, cheminformatics focuses on organizing chemical data, evaluating, and interpreting. In this time, we see many techniques like Quantitative Structure-Activity Relationships (QSAR), molecular property prediction and molecular similarity assessments, which are essential in predicting the features and activities of chemical compounds the development of drugs is streamlined and less dependent on extensive experimental testing.

The relation between graph, GNNs, deep learning, and cheminformatics shows great potential to improve and developed our comprehension and application of chemical data. By using GNNs capacity to model molecular structures as graphs, the Researchers can

improve molecular property prediction and enable more effective drug discovery and development processes.

Chapter 1:

An overview of the topic

Chapiter1

1 Introduction: 1

2 Overview 2

3 Background of the problem 2

4 Problem Statement 2

5 Objectives of the Research: 3

6 Scope of the Research 3

7 Theses Out line 3

8 Summary 4

CHAPTER I

1 Introduction:

Our world is constantly changing due to scientific and technical breakthroughs, which also improve our capacity to solve challenging issues in a variety of fields. The merger of cheminformatics, graph theory, and graph neural networks (GNNs) demonstrates this dynamic interplay in particular. These three disciplines, each with specialties in their own right, come together to spur innovation and address difficult issues, particularly in the areas of materials science and medication development.

Graph theory is a vital tool for many applications because it offers a strong mathematical foundation for modeling links and structures. It makes it possible to depict complicated systems as nodes (vertices) and connections (edges), which can be used to model a wide range of phenomena, including chemical structures and social networks. In order to comprehend and analyze the complex interactions inside a system, an abstract representation is essential.

Deep learning concepts are utilized by graph neural networks (GNNs) for the analysis of graph-structured data. Researchers can process and comprehend large volumes of data with intricate interdependencies using GNNs, something that traditional neural networks can find challenging. Given their exceptional performance in tasks such as node classification, link prediction, and graph classification, GNNs are highly valuable in relational data-related domains.

In contrast, cheminformatics addresses chemical problems by fusing informatics and chemistry. It entails using informational and computer methods to save, retrieve, evaluate, and forecast the characteristics and actions of chemical substances. Cheminformatics is especially effective in the drug development process because it can be used to predict the attributes, identify, and optimize the structures of possible drug candidates.

More precise molecular property predictions are made possible by the convergence of these domains, which is essential for a number of applications in materials science, healthcare, and other areas. The discovery of novel compounds with desirable therapeutic effects can be facilitated by the integration of graph theory, GNNs, and cheminformatics, greatly speeding up the process of developing new drugs.

CHAPTER I

In this chapter, develop a novel method of biological activity prediction based on deep learning Graph neural network (GNN) for Molecular property prediction, is presented.

The overview and the background of the problem, objectives, importance of the study, and the scope of this research is presented and discussed in this chapter.

2 Overview

The study focuses on using Graph Neural Networks (GNNs) for cheminformatics, specifically for chemical property prediction. Since GNNs are a form of deep learning model made to work with graph-structured data, modeling chemical structures with atoms represented as nodes and bonds as edges is a great use for them. The shortcomings of conventional cheminformatics techniques are addressed in this work by investigating the integration of GNNs to increase the precision and effectiveness of molecular property predictions.

3 Background of the problem

In cheminformatics, molecular property prediction plays a vital role influencing fields including drug research, materials science, and environmental science. Conventional methods frequently depend on expensive, time-consuming, and sometimes narrowly generalizable experimental techniques and specialized models. These approaches, which usually entail the development of molecular descriptors and the use of Quantitative Structure-Activity Relationships (QSAR) models, are beset by issues such as data shortages and class imbalance. GNNs present with their capacity to both naturally represent and learn from graph-structured data a viable a promising to these traditional techniques.

4 Problem Statement

While GNNs have a big potential for cheminformatics, but it necessary to do more research for determine how well these models predict chemical characteristics in comparison to more conventional techniques. The main problem is assessing how well GNNs perform in precisely simulating and forecasting the features of complicated

molecular structures while simultaneously tackling the issues of class imbalance and data scarcity present in cheminformatics datasets.

5 Objectives of the Research:

The main goal of this research is:

- Use deep learning Graph Neural Networks (GNNs) for chemical property prediction to create and verify a novel technique for biological activity prediction.
- Improve the accuracy and efficiency of molecular property predictions through the integration of graph theory, cheminformatics, and GNNs.
- Show how GNNs might transform the fields of materials science and medication development through more precise and effective predictions.
- Addressing the drawbacks of conventional methodologies, such as data scarcity and class imbalance through use of the sophisticated capabilities of GNNs to model and analyze complicated chemical structures.

6 Scope of the Research

The scope of this study is confined to the following in order to fulfil the above-mentioned objectives:

- The validation and evaluation of the quality of the prediction model proposed in this research will be done on different datasets that have been used to validate the generation of molecules based on Molecular Property Prediction.
- The databases aimed to be used in this study are only limited to chemical data from OPERA, and among various deep-learning models we use GNN for Molecular Property Prediction.

7 Theses Out line

This thesis consists of four main parts:

Chapter One: provides an introduction and brief overview of the project including problem statement, objective of the research, scope of the study, thesis outline and summary.

Chapter Two: literature reviews, begins in the first section with a basic of graph, also we talk about Deep learning and its background, and explain some of the relevant DL architectures like IA and Convolutional neural network.

In another title we talk about Graph Neural Network, architecture of GNN, application, the type, how it work ect …

In the second section we talk about an overview of chemoinformatics, then we present various type of molecular representations, Molecular similarity measurement, we talk also about Quantitative structure- activity relationships, Molecular property prediction and GNN for Molecular property prediction.

This chapter is concluded by with a discussion and summary of the applicability of the described methods mentioned methods of deep learning, there architectures and the best ways to improve the performance of these methods based on activities structures.

Chapter Three: Introduces python and his libraries, Google_Colab, that used in this project.

Chapter Four: We present in this chapter the implementation of our proposed approach for biological activity prediction, start by describing the data set, training and validation of the model. Then we explain the steps of our implementation, in the end we interpret and comment the results Obtained.

Finely Conclusion is the final chapter, which discusses and concludes this work, highlights the contribution and results of our work, and provides suggestions and recommendations for future research.

8 Summary

In this chapter, we try to give a broad overview of the problems involved in the molecular generation and applied GNN for Molecular property prediction which is the subject of our work.

This chapter serves as an introductory to the research problem addressed in this study.

The goal, objectives, the scope, and the outline of this work were presented.

Chapter 2:

LITERATURE REVIEW

Chapter 2

1 Introduction 5
2 Basic of the graph 7
2.1 Whatis the graph 7
2.2 Graph components: 7
2.2.1 The node: 7
2.2.2 The edge: 7
2.3 Types of Graphs: 7
2.4 Graph representations: 10
3 Deep Learning 12
3.1 Artificial Intelligence (AI): 12
3.1.1 Definition: 12
3.1.2 Types of AI: 13
3.2 Machine Learning (ML): 13
3.2.1 Definition: 13
3.2.2 Type: 13
3.3 Deep Learning (DL) 15
3.3.1 Definition: 15
3.3.2 Application of DL: 16
3.4 Differences between AI, machine learning and deep learning: 16
3.5 Convolutional neural network (CNN): 18
3.5.1 Definition: 18

3.5.2 CNN Architecture: 18
3.5.3 Advantages of Convolutional Neural Networks (CNNs): 19
3.5.4 Disadvantages of Convolutional Neural Networks (CNNs): 19
4 Graph neural networks 19
4.1 Graph neural networks definition: 19
4.2 The Type of Graph Neural Networks Architecture: 20
4.2.1 Graph Convolutional Network (GCN): 20
4.2.2 Graph Attention Networks (GAT): 21
4.2.3 GraphSAGE: 23
4.2.4 Graph Recurrent Neural Networks (GRNNs): 23
4.2.5 Equation of GNN Models: 24
4.3 Application of Graph Neural Networks: 24
4.4 How GNN works? 25
4.5 Where we can use GNN? 26
5 Definition of Chemoinformatics: 29
6 Molecular Representations: 30
7 Molecular similarity measurement: 31
7.1 Introduction: 31
7.2 Definition: 32
7.3 Similarity Coefficients: 32
7.4 Fingerprint similarity: 34
8 Quantitative structure- activity relationships: 35
8.1 Definition: 35
8.2 The Steps of QSAR study: 35
8.3 Advantages and disadvantages of QSAR: 36
8.3.1 Advantages: 36
8.3.2 Disadvantages: 36
9 Molecular property prediction: 36
9.1 Definition: 36
9.2 Where we can use MPP: 36
9.3 Challenges Molecular Property Prediction: 37
10 Graph neural networks for molecular property prediction: 37
11 Conclusion: 39

1 Introduction

A graph is a mathematical structure made up of vertices, or nodes, and edges. It is denoted as G = (V, E). Edges indicate the links or interconnections between these nodes, whereas nodes stand for entities or data items. Graphs can be limited or infinite in terms of the number of vertices and edges, and they can be directed or undirected based on whether the edges have a specified direction. Adjacency matrices and Laplacian matrices are two examples of the many representations that are used to show graphs for various analytical uses.

Graph Neural Networks (GNNs) are a class of deep learning techniques designed to process graph-structured data. They support tasks such as node, edge, and graph-level prediction. Common GNN architectures include graph convolutional networks (GCNs), graph attention networks (GATs), and graph recurrent neural networks (GRNNs). These networks have a wide range of applications in areas such as recommender systems, healthcare, network security, and social network analysis due to their ability to effectively model complex relational data structures.

Chemicalinformatics, also name for hemoinformatics, is an interdisciplinary field that sits at the nexus of information science, computer science, and chemistry. It is centered on organizing, deciphering, and controlling chemical data in order to turn it into insightful knowledge. The field focuses on using computers to help solve chemical challenges, especially in the creation and discovery of new drugs. Developing molecular descriptors—mathematical representations of molecules—and measuring molecular similarity—an assessment of the structural or physicochemical similarities between chemical substances—are important aspects of chemoinformatics.

Quantitative structure-activity relationships (QSAR), a modeling approach that connects a chemical substance's structural characteristics to its biological activities or other aspects, is also included in the field of chemoinformatics. In order to anticipate compound activity, facilitate medication development, and lessen the need for lengthy experimental testing, QSAR models are essential. Class imbalance and a lack of data make molecular property prediction difficult even with its promise.

All things considered, the subject of chemoinformatics is dynamic and constantly

changing, contributing greatly to our knowledge and utilization of chemical data in a variety of scientific fields. Future developments in drug discovery, material design, and the use of artificial intelligence and machine learning into chemical research appear promising.

In recent decades, the problem of predicting molecular attributes has grown in importance and difficulty. Conventional approaches to forecast these qualities frequently involve specialized experiments, expensive and time-consuming R&D, and specialized models with limited generalizability.

Because graph neural networks (GNNs) can represent graph-structured data, they have gained popularity as a solution. GNNs depict molecules as topological graphs with atoms as nodes and bonds as edges, which allow them to predict molecular attributes with high accuracy. This graph-based method is comparable to supervised graph classification issues since it readily collects molecular structure information.

2 Basic of the graph

2.1 Whatis the graph

A graph G = (V, E) is defined as a collection of two sets V and E , where V is a finite set of nodes and E ⊆V × V is the set of edges.[1]

2.2 Graph components:

2.2.1 The node:

The nodes are also known as vertices, and these are the entities or data points in the graph. Simple examples of nodes are people, places, and things. These are associated with the features or attributes that describe the node and these are known as node features. These features vary according to the type of graphical network. [2]

2.2.2 The edge:

Edges are the connections between the nodes, and these are also known as the connections, links, or relationships of the nodes. Edges may be directional or unidirectional in nature, and these play a vital role in the connection of one node to the other. The unidirectional nodes represent the relationship of one node to the other, and the undirected edges represent the bidirectional relationship between the nodes. [2]

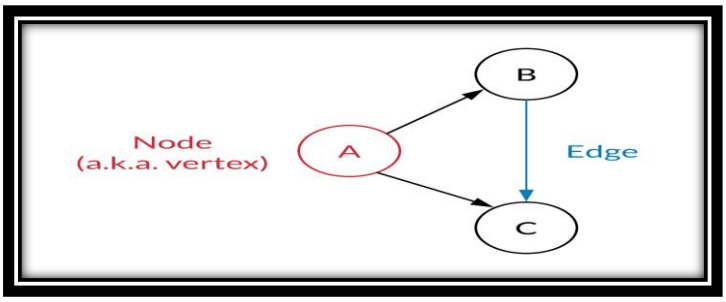


Figure 1: Graph components.

2.3 Types of Graphs:

2.3.1 Finite & Infinite Graph:

- If the number of vertices and edges in the graph G= (V, E) is finite, so we named this graph as a **finite graph**.

- If the graph G= (V, E) has an infinite number of vertices and edges, it is referred to as an **infinite graph**. [3]

2.3.2 Directed & undirected graphs:

- A **directed graph** is a graph that in which all edges are associated with a direction.

- An **undirected** graph is a graph in which all edges do not have a direction. [3]

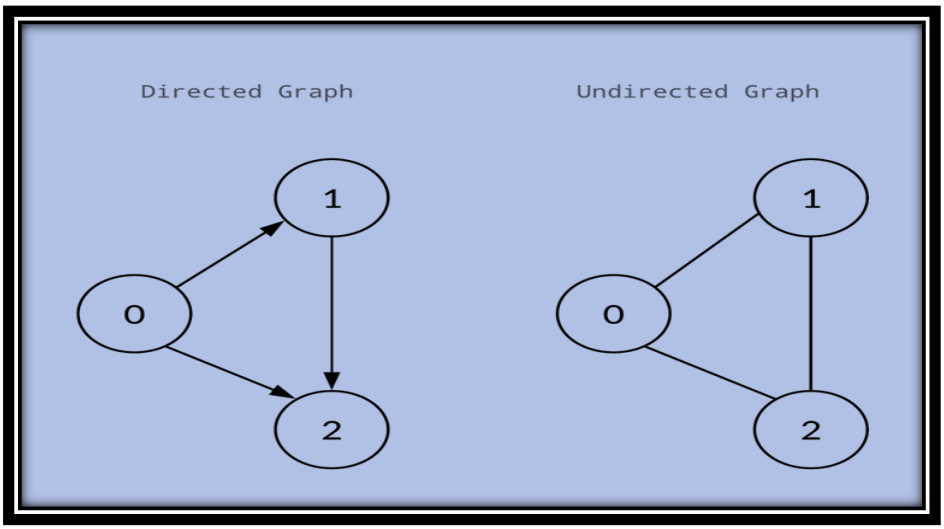


Figure 2: Directed and undirected graphs**.**

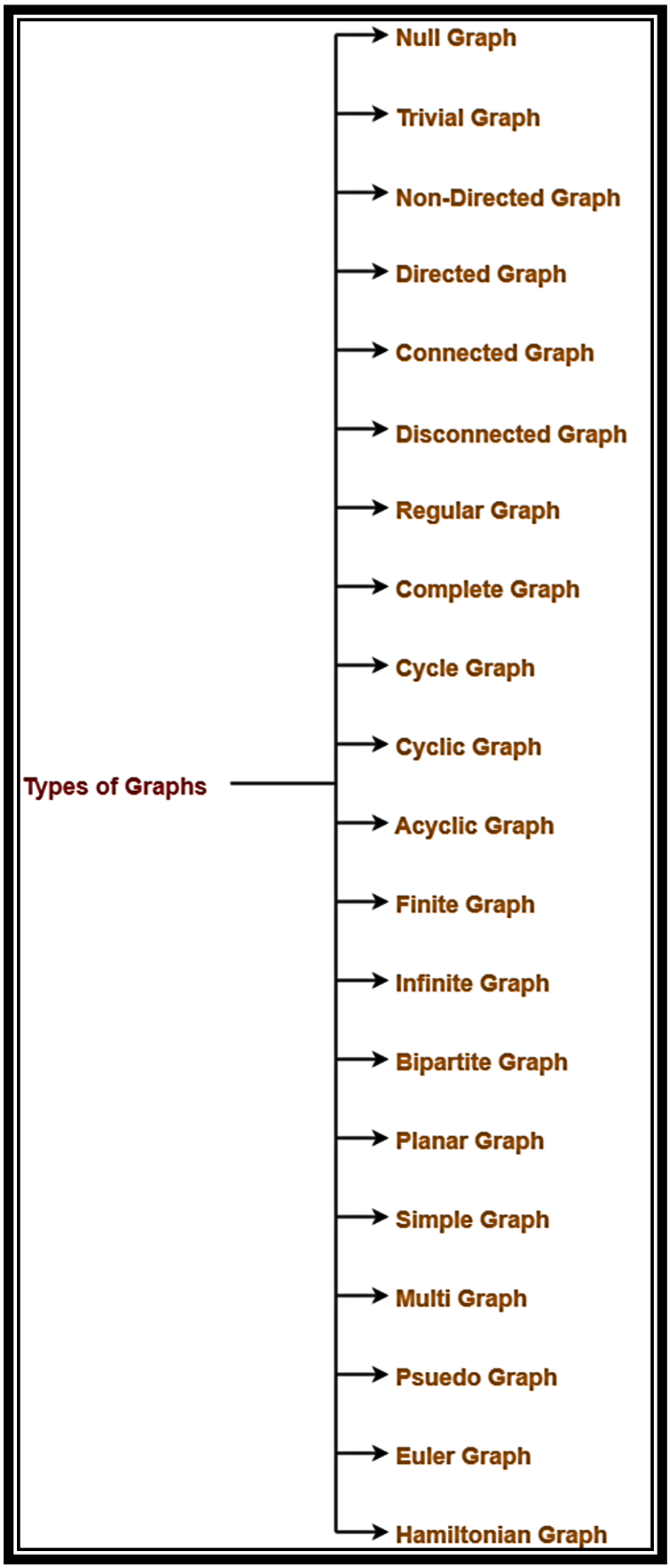


Figure 3: Type of Graph.

2.4 Graph representations:

2.4.1 Adjacency matrix:

For a given graph G = {V, E}, the corresponding adjacency matrix is denoted as :

A $\in\{0,1\}^{N\times N}$. The i, j-th entry of the adjacency matrix A, indicated as , A_{ij} represents the connectivity between two nodes v_i and v_i. More specifically,

A_{ij} = 1 if v_i is adjacent to v_j, otherwise A_{ij} = 0. [4]

$$A_{ij} = \begin{cases} 1 & \text{if } \{v_i; v_j\} \in E \text{ and } i \neq j \ ; \\ 0 & \text{otherwise.} \end{cases}$$

Example:

✓ **Representation of Undirected Graph to Adjacency Matrix:**

Initially, the entire Matrix is initialized to 0. If there is an edge from source to destination, we insert 1 to both cases (adjMat[destination] and adjMat[destination]) because we can go either way. [5]

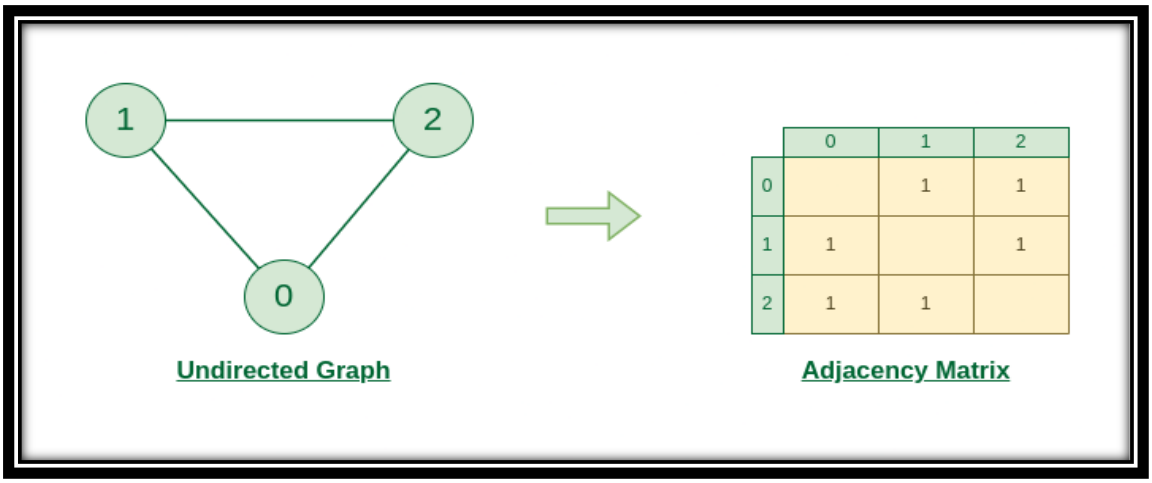


Figure 4: Representation of Undirected Graph to Adjacency Matrix.

2.4.2 Representation of Directed Graph to Adjacency Matrix:

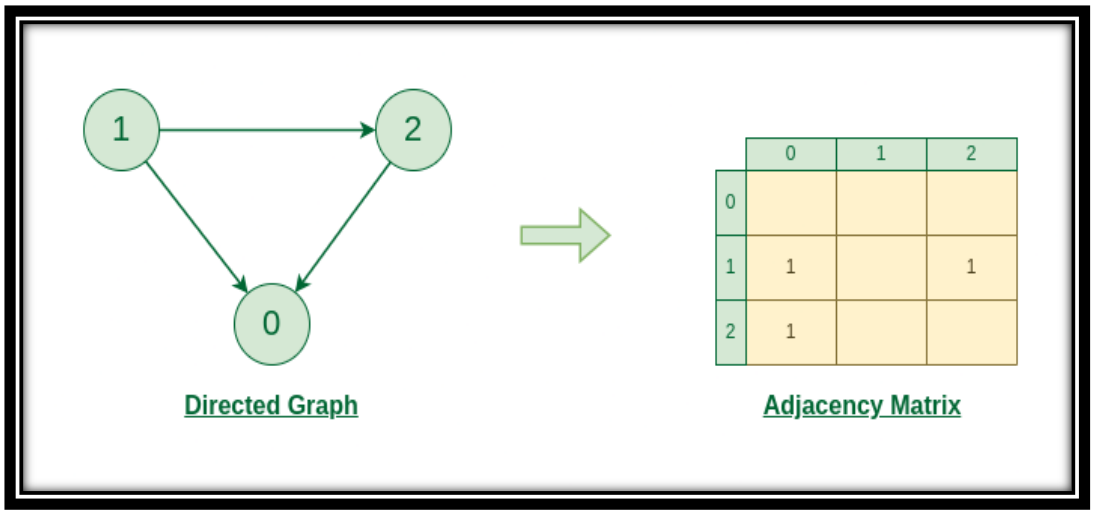


Figure 5: Representation of Directed Graph to Adjacency Matrix.

2.4.3 Degree matrix:

For a graph with G = (V, E) n-vertices, its degree matrix D ∈ $R^{n\times n}$ is a diagonal matrix, where:

$D_{ii} = d(v_i)$. [6]

2.4.4 Laplacian matrix:

Given a simple undirected graph G = (V, E), we can be definedits Laplacian matrix L∈ $R^{n\times n}$ as:

L=D – A.

With L: the Laplacian matrix

And where D ,A are the node degree matrix and adjacency matrix of the graph G,

Now we have the elements:

$$L_{ij} = \begin{cases} d(v_i) & \text{if i = j ;} \\ -1 & \text{if } \{v_i; v_j\} \in \text{E and i} \neq \text{j ;} \\ 0 & \text{otherwise .} \end{cases} \quad \text{. [6]}$$

2.4.5 Symmetric normalized Laplacian:

The symmetric normalized Laplacian $\boldsymbol{L^{sym}}$ was defined as:

$$\boldsymbol{L^{sym} = D^{-\frac{1}{2}} L D^{-\frac{1}{2}}}$$

$$\boldsymbol{= I - D^{-\frac{1}{2}} A D^{-\frac{1}{2}}}\ .$$

The elements can be computed by:

$$L_{ij}^{sym} = \begin{cases} \mathbf{1} & \text{if i} = \text{j } and d(v_i) \neq 0\text{ ;} \\ -\dfrac{\mathbf{1}}{\sqrt{\boldsymbol{d}(\boldsymbol{v_i})\boldsymbol{d}(\boldsymbol{v_j})}} & if\ \{v_i, v_j\} \in \text{ E and i } \neq \text{ j ;} \\ \mathbf{0} & \text{otherwise.} \end{cases}$$ [6]

2.4.6 Random walk normalized Laplacian:

It has the following definition:

$$\boldsymbol{L^{rw} = D^{-1}L = I - D^{-1}A}.$$

The method of computing the elements is**:**

$$\boldsymbol{L_{ij}^{rw}} = \begin{cases} \mathbf{1} if \boldsymbol{i = j and d(v_i)}; \\ \mathbf{0}\text{otherwise;} \\ -\dfrac{\mathbf{1}}{\boldsymbol{d(v_i)}} \quad \boldsymbol{if}\ \{\boldsymbol{v_i, v_j}\} \in \text{ E and i } \neq \text{ j .} \end{cases}$$

2.4.7 Incidence matrix:

For a simple graph G = (V, E) with n-vertices and m-edges,the corresponding incidence matrix is $\mathbf{M} \in \boldsymbol{R^{n\times m}}$ **,** where

$$M_{ij} = \begin{cases} 1 & if \exists ks.te_j = \{v_i, v_k\}\text{ ;} \\ -1 & if \exists ks.te_j = \{v_k, v_i\}; \\ 0 & \text{otherwise.} \end{cases}$$

And for an undirected graph, the corresponding incidence matrix is:

$$M_{ij} = \begin{cases} 1 & if \quad \exists ks.te_j = \{v_i, v_k\} \\ 0 & \text{otherwise.} \end{cases}$$ [6]

3 Deep Learning

3.1 Artificial Intelligence (AI):

3.1.1 Definition:

Artificial intelligence commonly referred to as AI, is the process of imparting data, information, and human intelligence to machines. The main goal of Artificial Intelligence

is to develop self-reliant machines that can think and act like humans. These machines can mimic human behavior and perform tasks by learning and problem-solving. Most of the AI systems simulate natural intelligence to solve complex problems. [7]

3.1.2 Types of AI:

- Narrow AI.
- Artificial General Intelligence (AGI).
- Artificial Super intelligence (ASI).
- Reaactive machine AI.
- Limited Memory AI.
- Theory of mind AI.
- Self-Aware AI. [8]

3.2 Machine Learning (ML):

3.2.1 Definition:

Machine learning (ML) is a subfield of artificial intelligence (AI) that uses algorithms trained on data sets to create self-learning models capable of predicting outcomes and classifying information without human intervention. [9]

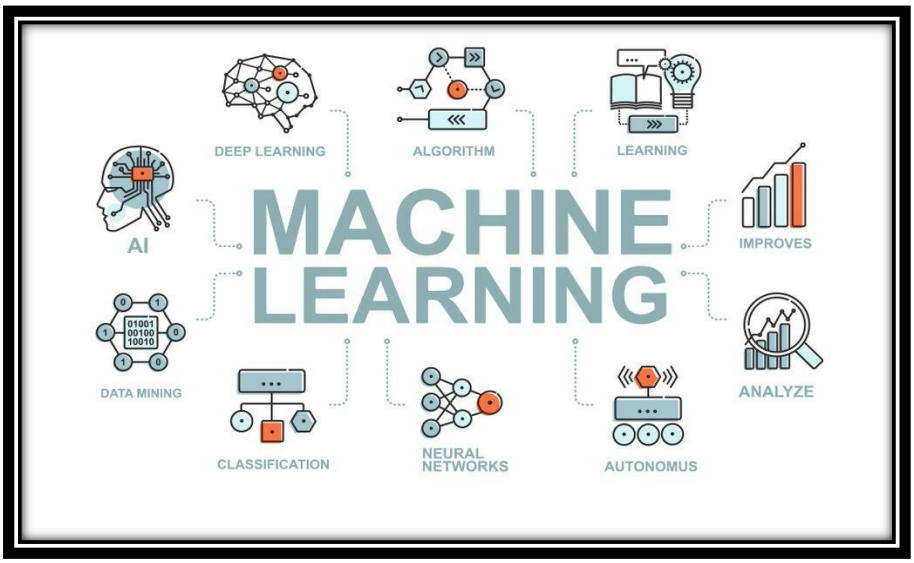


Figure 6: Machine Learning

3.2.2 Type:

- **Supervised learning:**
  - Trained on labeled datasets with both input and output parameters.

- Examples include image classification (differentiating between cats and dogs) and predicting categorical target variables (e.g., spam vs. not spam).
- Algorithms: Logistic Regression, Support Vector Machine, Random Forest, Decision Tree, K-Nearest Neighbors (KNN), and Naive Bayes. [10]

- **Unsupervised learning:** also known as unsupervised machine learning, uses machine learning (ML) algorithms to analyze and cluster unlabeled data sets. These algorithms discover hidden patterns or data groupings without the need for human intervention. [11]
- **Semi-supervised learning:** Semi-supervised learning is a type of machine learning .It uses a combination of labeled and unlabeled data to train artificial intelligence (AI) models for classification and regression problems.

  In semi-supervised learning, the model makes use of both the relatively large amount of unlabeled data and the labeled data, which supply the required context and accurate predictions, to enhance its performance.

  This method works particularly well in situations where it is costly or difficult to get enough labeled data, but unlabeled data is easily accessible. [12] [13]

- **Reinforcement Learning:** Reinforcement machine learning algorithm is a learning method that interacts with the environment by producing actions and discovering errors. Trial, error, and delay are the most relevant characteristics of reinforcement learning. In this technique, the model keeps on increasing its performance using Reward Feedback to learn the behavior or pattern. These algorithms are specific to a particular problem. [10]

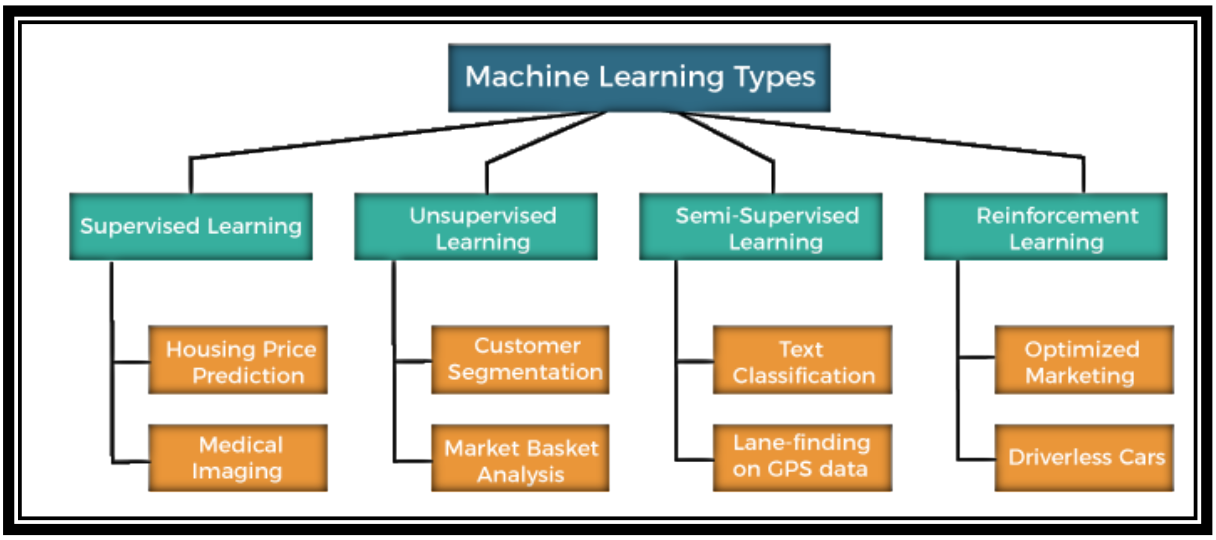


Figure 7: Types of Machine leaning.

3.3 Deep Learning (DL)

3.3.1 Definition:

Deep learning is a type of machine learning that uses artificial neural networks to learn from data, [14] it has three layers:

- Input layer: Data enters through the input layer.
- Hidden layers: Hidden layers process and transport data to other layers.
- Output layer: The final result or prediction is made in the output layer. [15]

3.3.2 Application of DL:

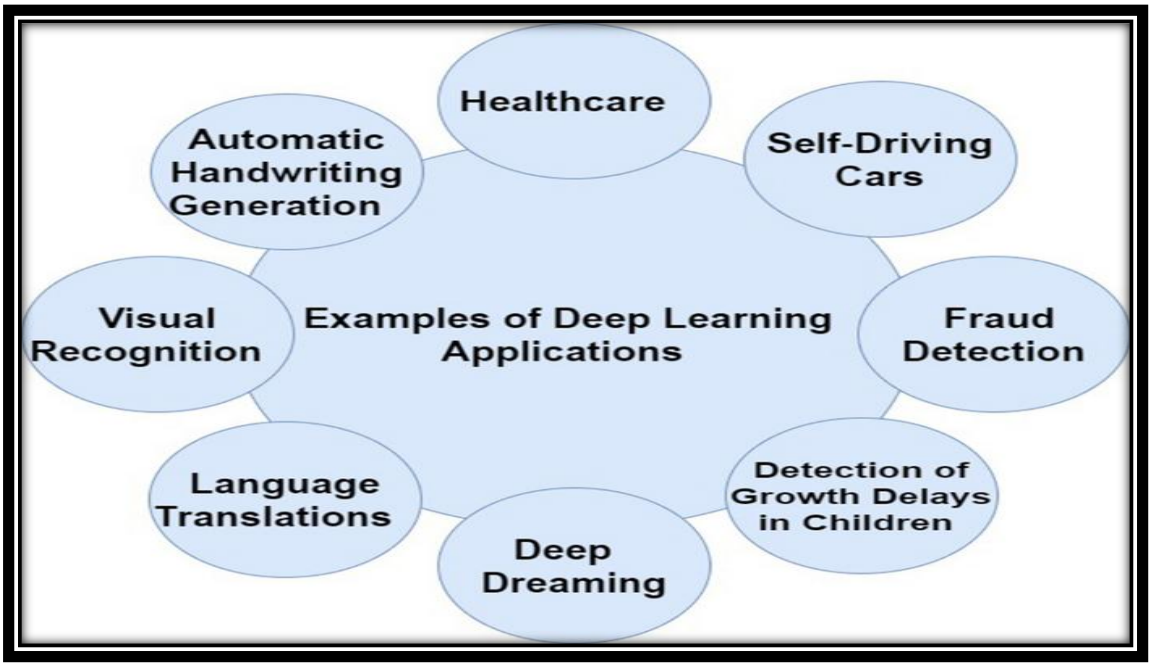


Figure 8: Application of DL

3.4 Differences between AI, machine learning and deep learning:

Artificial Intelligence, Machine Learning, and Deep Learning have become the most talked-about technologies in today's commercial world as companies are using these innovations to build intelligent machines and applications. [7]

Table 1: differences between Artificial Intelligence, Machine Learning and Deep Learning:

Artificial Intelligence	Machine Learning	Deep Learning
AI, is a subfield of computer science, that was created in the 1960s with the aim of creating machines that are intelligent, basically which can do any tasks humans are capable of doing too, but taking lesser time than a human might take.	One of the subfields of AI - in the past, it covered all types of data science algorithms, supervised, unsupervised, segmentation, classification, or regression but presently this term refers to unsupervised pattern recognition.	It is a proper subset of Machine Learning, essentially a set of techniques that help you to parameterize deep neural network structures i.e. neural networks numerous layers and parameters.
Making machines intelligent may or may not need high computational power as it depends on the nature of the task that needs to be learnt by the machine or in better terms, needs to be automated.	These algorithms can work easily on normal low performance computers without GPUs as there isn't a need for lot of memory storage and saving of trained models.	Algorithms are dependent on high performance hardware components that include GPUs (and now TPUs too) that can efficiently optimize operations like matrix multiplication.
Based on the triviality or complexity of the task, a suitable algorithm is chosen so as to minimise the overall cost of solving the problem in as less time as possible.	Generally, they follow the Divide and Conquer concept i.e. breaking down the problem into different parts, solving them individually and combining them to get the output.	Based on solving the problem end-to-end, training the model with high level features derived from low level features.
Tasks can be anything that has a solution in reality based on the appropriate type and amount of data, and the features may or may not be available. Machines can solve a problem only if they have the right kind of data as input.	Most of the required features need to be identified and coded manually therefore take more time in the testing phase than the training phase of the model.	Algorithms try to learn high-level features from data based on the low-level features identified by them therefore take more time in the training phase as compared to testing phase.
Problems to be solved by AI need data to work on, maybe a lot or very less, which needs to be found, cleaned, checked for outliers and then be used to obtain results for the problem.	Algorithms perform well and even better than Deep Learning models when the amount of data available for training and testing is less.	These models necessarily need huge amounts of data to learn from it and give better accuracy in their predictions.
AI is a vast field containing several algorithms that may not necessarily be easy to understand based solely on logical thinking. Sometimes, the only way is to mathematically understand particular concepts.	Understanding how certain results are given by these algorithms is very easy as they work on a set of defined rules.	It is very difficult to understand why the predicted values are the way they are. We can derive the reason behind the predictions mathematically but that's very complicated.

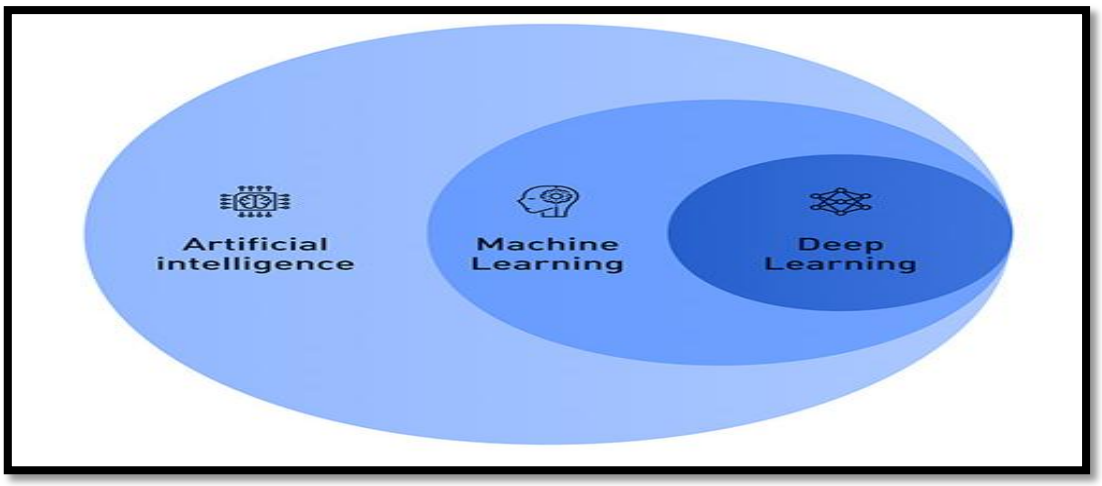


Figure 9: Represents The Relation Between AI, ML and Deep Learning.

3.5 Convolutional neural network (CNN):

3.5.1 Definition:

CNN is a type of deep learning model for processing data that has a grid pattern, such as images, which is inspired by the organization of the animal visual cortex and designed to automatically and adaptively learn spatial hierarchies of features, from low- to high-level patterns. [16]

3.5.2 CNN Architecture:

The CNN architecture consists of several kinds of layers: Convolutional layer, pooling layer, fully connected layer.

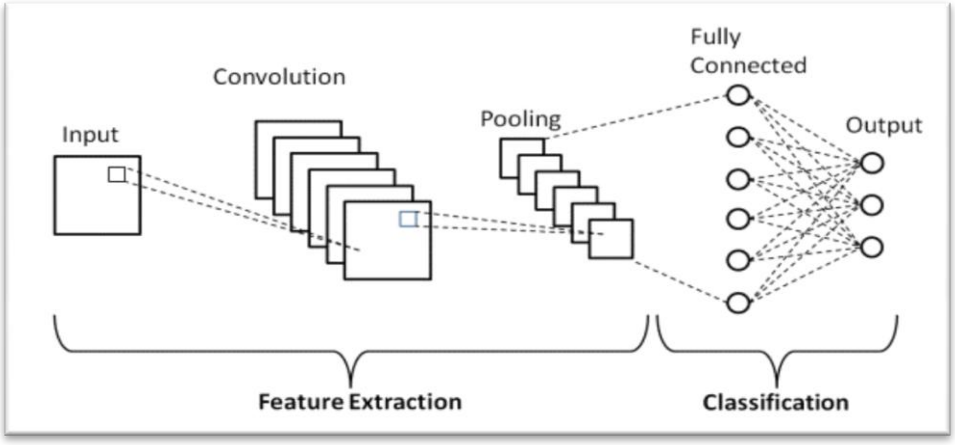


Figure 10: A Simple Architecture of CNN.

- Convolution layer: In each convolutional layer, there are several filters (kernels) k of size n × n × q. In this case, n should be less than the input image, but q may be the same size as C or less. [17]
- Pooling layer: Each feature map in the sub sampling layers is down sampled to reduce the number of network parameters, expedite training, and hence prevent over fitting. [17]
- Fully connected layers: In normal neural networks, the latter stage layers are often fully connected. These layers create the high-level abstraction from the data using the low- and mid-level characteristics that came before. [17]

3.5.3 Advantages of Convolutional Neural Networks (CNNs):

- Good at detecting patterns and features in images, videos, and audio signals.
- Robust to translation, rotation, and scaling invariance.
- End-to-end training, no need for manual feature extraction.
- Can handle large amounts of data and achieve high accuracy. [18]

3.5.4 Disadvantages of Convolutional Neural Networks (CNNs):

- High computational requirements.
- Needs large amount of labeled data.
- Large memory footprint.
- Interpretability challenges.
- Limited effectiveness for sequential data.
- Tend to be much slower.
- Training takes a long time. [19]

4 Graph neural networks

4.1 Graph neural networks definition:

Graph Neural Networks (GNNs) are a class of deep learning methods designed to perform inference on data described by graphs.

GNNs are neural networks that can be directly applied to graphs, and provide an easy way to do node-level, edge-level, and graph-level prediction tasks.

GNNs can do what Convolutional Neural Networks (CNNs) failed to do. [20]

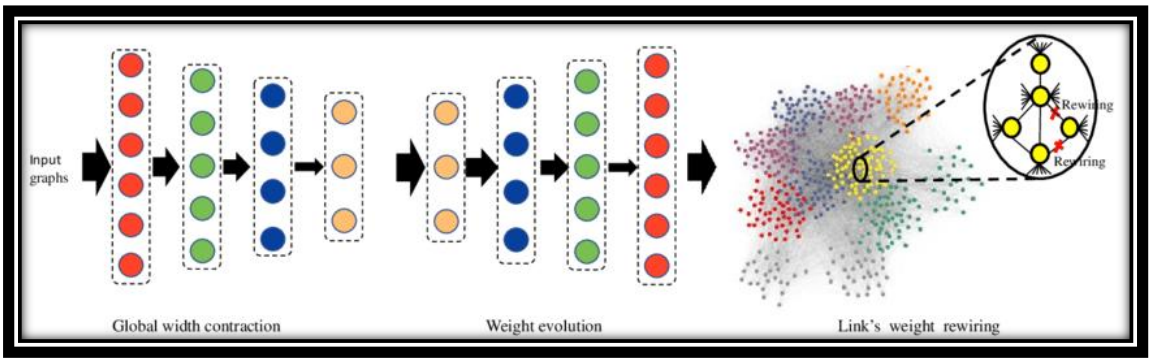


Figure 11: A graph neural network architecture design.

4.2 The Type of Graph Neural Networks Architecture:

The popular architectures of graph neural networks are:

4.2.1 Graph Convolutional Network (GCN):

- **Definition :**

GCNs are one of the earliest and most popular types of GNNs. They generalize the convolution operation from traditional CNNs to graph-structured data by aggregating features from neighboring nodes. [21]

- **working of GCN :**

- Initialization
- Convolution Operation
- Weighted Aggregation
- Activation function and learning weights
- Stacking Layers
- Task-Specific Output. [22]

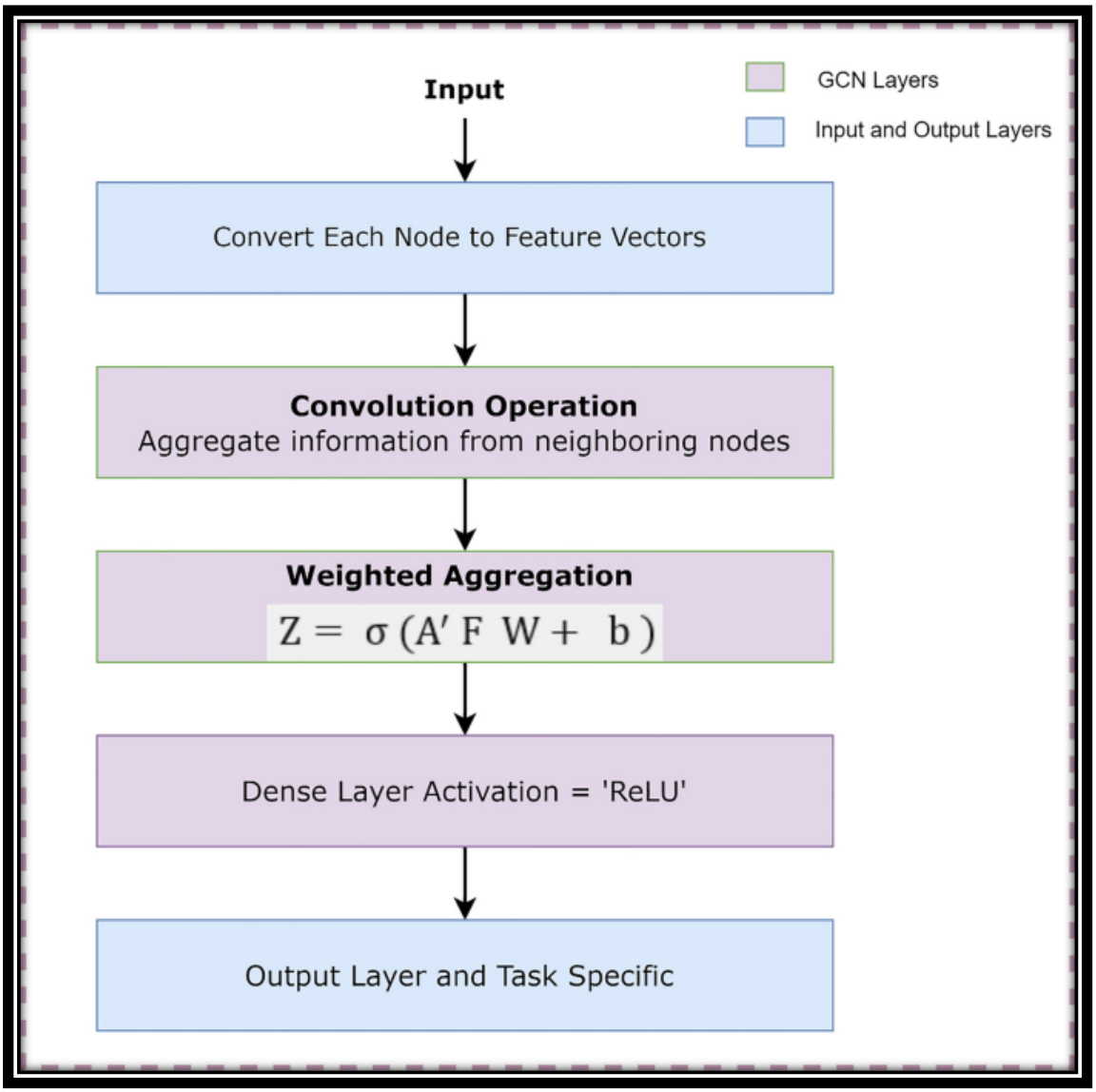


Figure 12: working GCN.

4.2.2 Graph Attention Networks (GAT):

- **Definition :**

GATs leverage the attention mechanism to assign different importance weights to neighboring nodes during the feature aggregation process, enabling the model to focus on the most relevant information. [23]

- **Working of GAT :**

- Initialization
- Self-Attention Mechanism and Attention Computation
- Weighted Aggregation

- Multiple Attention Heads and Output Combination
- Learning Weights and Stacking Layers
- Task-Specific Output.[22]

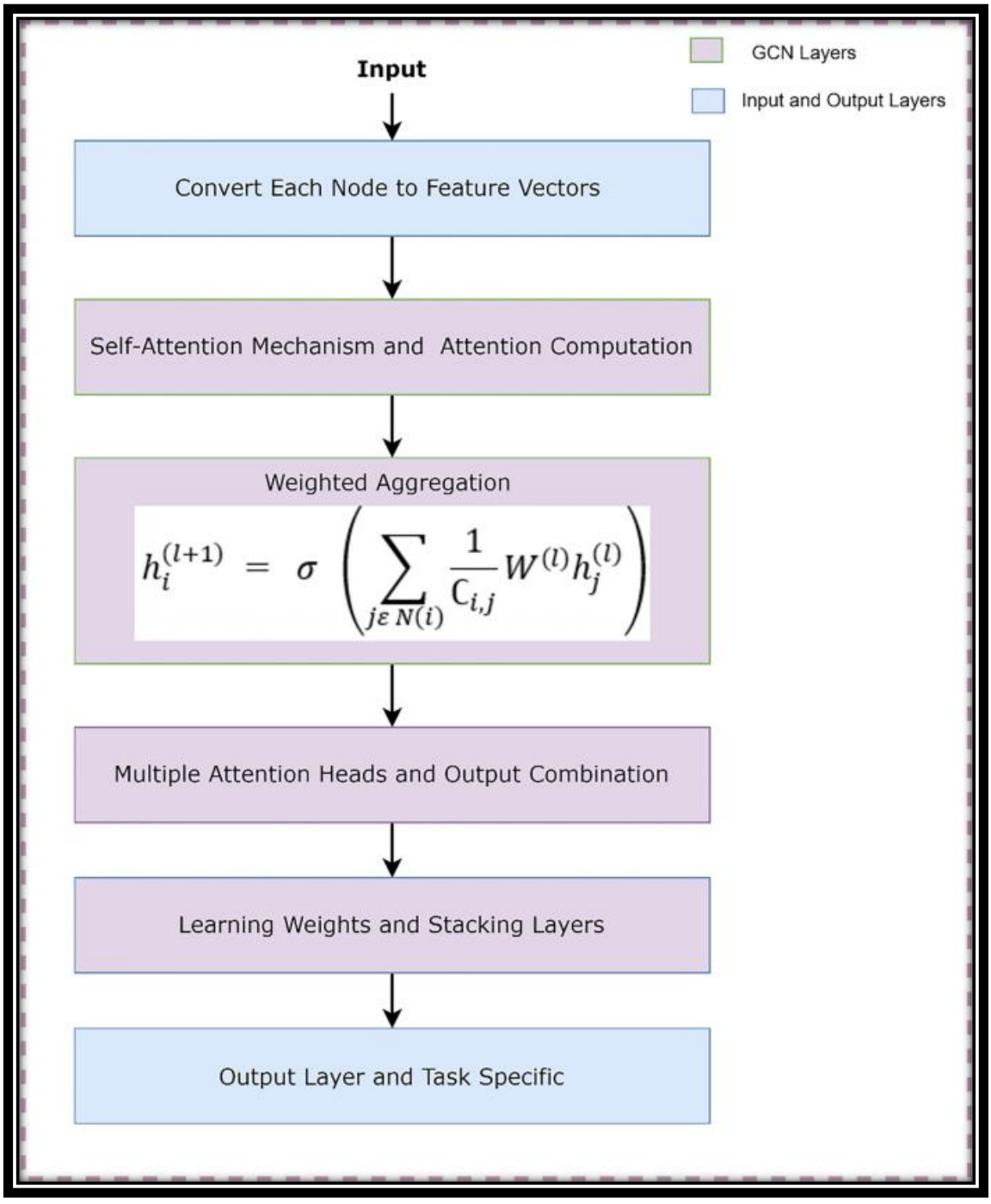


Figure 13: Working of GAT.

4.2.3 GraphSAGE:

- **Definition :**

GraphSAGE is an inductive framework that learns to aggregate feature information from a node's local neighborhood for generating node embeddings. It can be applied to previously unseen graphs, making it suitable for inductive tasks. [24]

- **Working of graphSAGE model:**

- Initializes the eigenvectors of all nodes in the input graph
- For each node, get its sampled neighbor nodes
- The aggregation function is used to aggregate the information of neighbor nodes
- And combined with embedding, update the same by a non-linear transformation embedding Express. [22]

4.2.4 Graph Recurrent Neural Networks (GRNNs):

- **Definition :**

GRNNs use recurrent neural networks (e.g., LSTMs or GRUs) to propagate information along the edges of a graph, making them suitable for tasks involving sequential or temporal data on graphs. [25]

$$x^k = T\left(x^{k-1}\right), \mathrm{k} \in (1, n)$$

- **Architecture of Recurrent Graph Neural Network:**

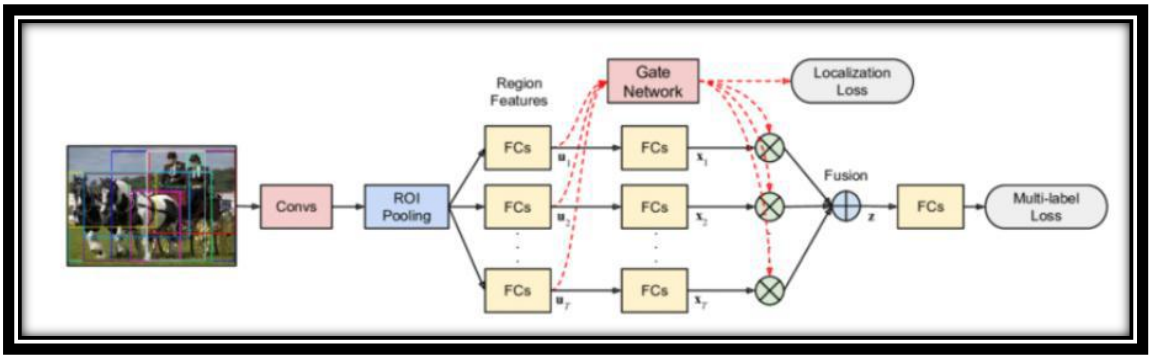


Figure 14: Architecture of Recurrent Graph Neural Network**.**

4.2.5 Equation of GNN Models:

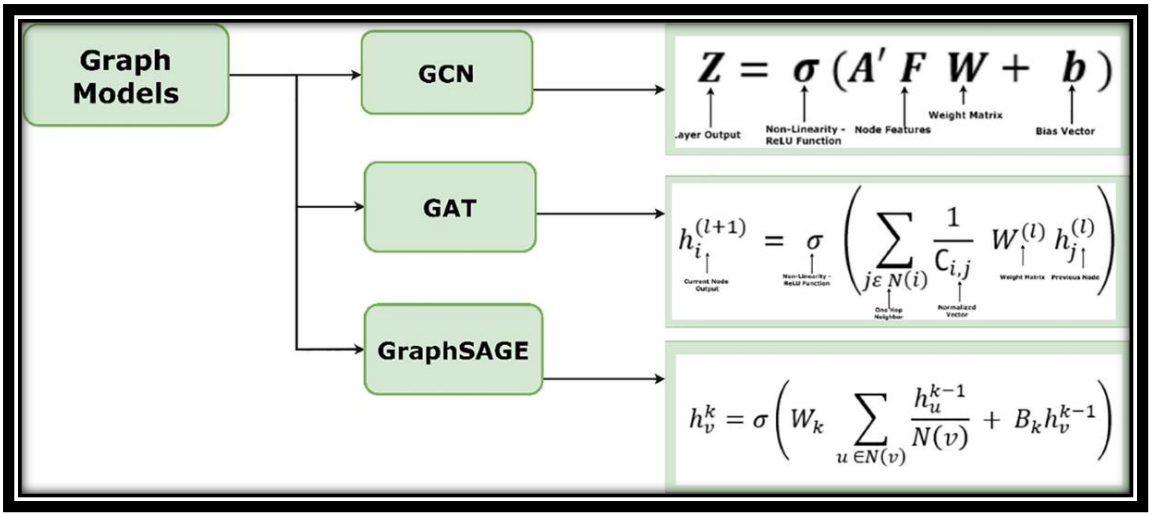


Figure 15: The Equations of GNN Models.

4.3 Application of Graph Neural Networks:

Graph-structured data is present everywhere. The problems that GNNs resolve can be classified into these categories:

- Graph Classification: we use this to classify graphs into various categories. Its applications are social network analysis and text classification.
- Node Classification: this task uses neighboring node labels to predict missing node labels in a graph.
- Link Prediction: predicts the link between a pair of nodes in a graph with an incomplete adjacency matrix. It is commonly used for social networks.
- Community Detection: divides nodes into various clusters based on edge structure. It learns from edge weights and distance and graph objects similarly.
- Graph Embedding: maps graphs into vectors, preserving the relevant information on nodes, edges, and structure.
- Graph Generation: learns from sample graph distribution to generate a new but similar graph structure. [26]

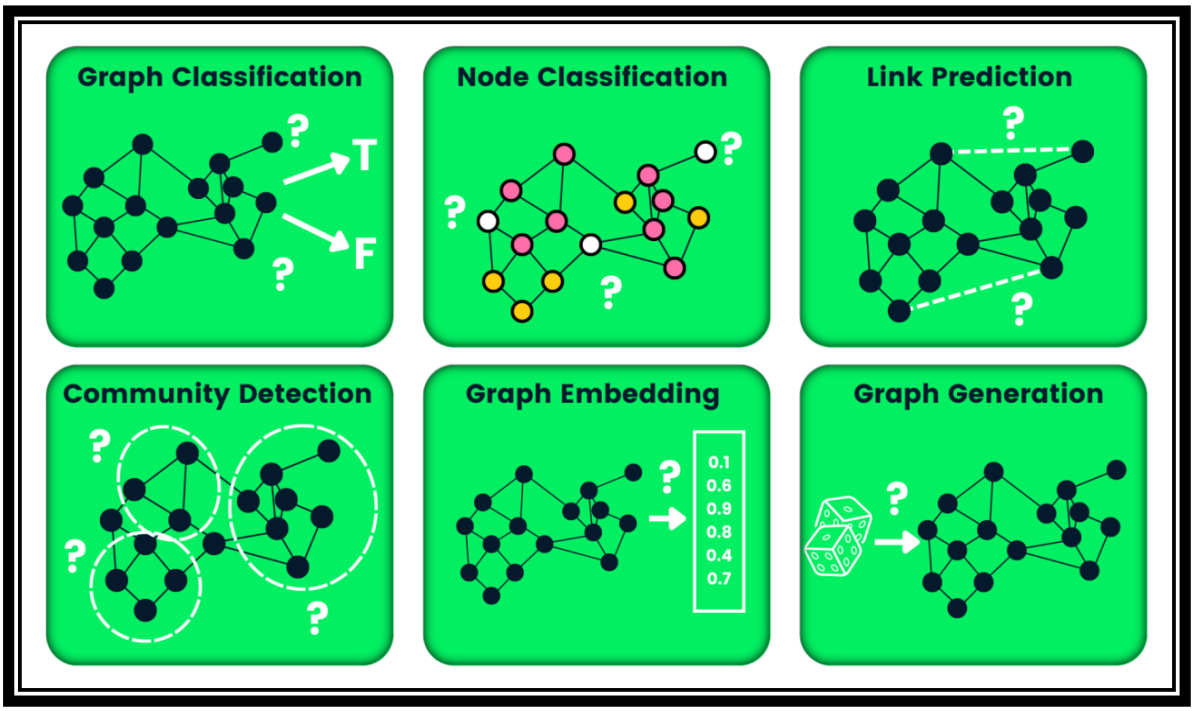


Figure 16: Difference between graph and node classification.

4.4 How GNN works?

- We start with the initial representation of the graph as input.
- GNN consists of multiple layers k.
  - In the first layer, the network learns the representation of the immediate neighbors of the nodes.
  - In 2nd layer, the network learns the representation of neighbors of the neighbors, and so on.

  The number of layers K is a hyper parameter and can be any value.
- During the training process,
  - In the forward pass, the layers calculate the representation of the nodes of the graphs.
  - It then calculates the loss, i.e. the difference between the current representation and the actual representation, and backpropagates the error.
  - In the next iteration, the network adjusts the node representation to minimize the error further, and so on till the error cannot be reduced further.
- Once the GNN is trained and it has learned the embeddings /representation of the graph nodes/edges, it can be then used to solve graph learning problems like node classification, link prediction, graph classification…. [27]

4.5 Where we can use GNN?

Graph Neural Networks (GNNs) are used to model and analyze data structures with relationships and can be applied to a wide range of domains. Here are some specific use cases:

4.5.1 Social Network Analysis:

Social Network Analysis: GNNs are used to analyze social media networks to predict friendships and interactions between users, detect communities, or identify influential nodes (influencers) by analyzing the connections between individuals. [28]

4.5.2 Recommendation Systems:

GNNs are widely used for personalized recommendations in e-commerce, social media, and content platforms, tailoring suggestions based on user interactions. [28]

4.5.3 Medicine:

Graph neural networks can analyze patient data, such as their age and gender, and find trends that correspond to those factors. This allows medical professionals to identify trends in their diagnoses and treatments that are specific to various patient demographics. [29]

4.5.4 Geolocation:

GNNs are able to improve the capacity for geolocation software, which includes such navigation applications as Google Maps, to inform drivers of their ETA.

This is down to the fact that graph neural networks can process multiple nodes and cross-check those factors. [29]

4.5.5 Natural Language Processing (NLP):

A wide range of NLP tasks can benefit from the use of GNN, including sentiment classification, text classification, and sequence labelling, to name just a few. In NLP, they are employed because of their convenience. Social Network Analysis uses them to forecast similar postings and provide users with relevant content recommendations. [30]

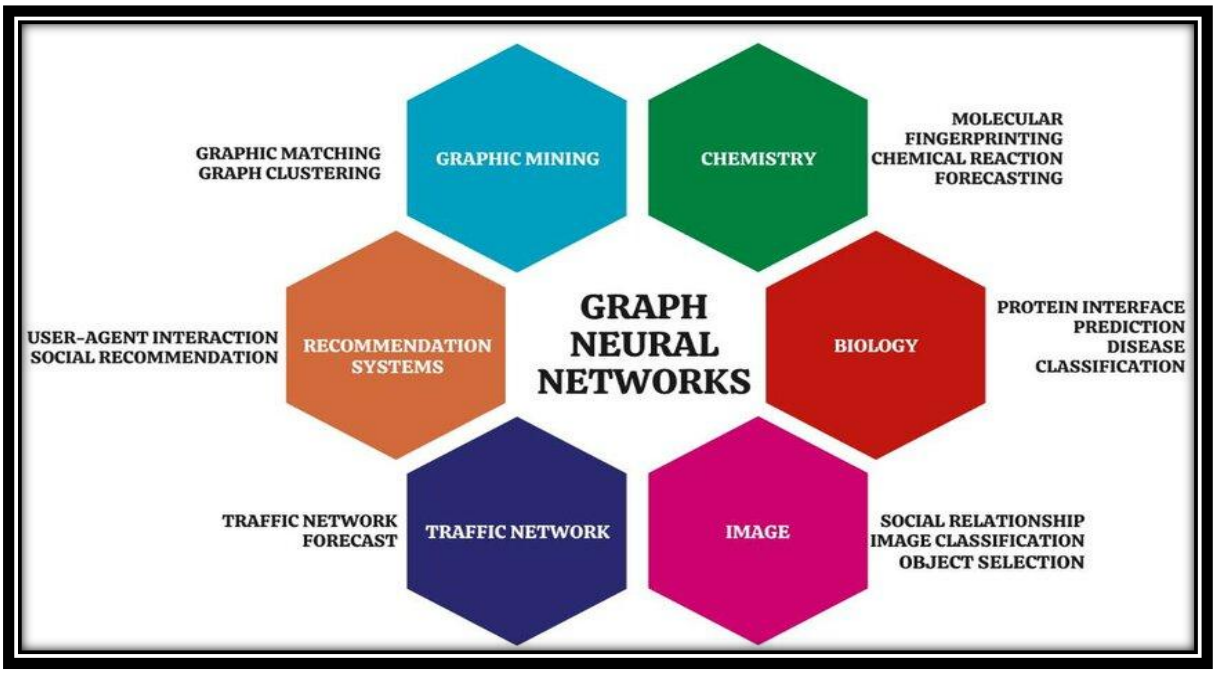


Figure 17: GNN Application**.**

4.5.6 More applications of GNNs:

Table 2: Application of Graph Neural Network:

Area	Application	Algorithm	Deep Learning Model
Text	Text classification	GCN	Graph Convolutional Network
		GAT	Graph Attention Network
		DGCNN	Graph Convolutional Network
		Text GCN	Graph Convolutional Network
		Sentence LSTM	Graph LSTM
	Sequence Labeling (POS, NER)	Sentence LSTM	Graph LSTM
	Sentiment classification	Tree LSTM	Graph LSTM
	Semantic role labeling	Syntactic GCN	Graph Convolutional Network
	Neural machine translation	Syntactic GCN	Graph Convolutional Network
		GGNN	Gated Graph Neural Network
	Relation extraction	Tree LSTM	Graph LSTM
		Graph LSTM	Graph LSTM
		GCN	Graph Convolutional Network
	Event extraction	Syntactic GCN	Graph Convolutional Network
	AMR to text generation	Sentence LSTM	Graph LSTM
		GGNN	Gated Graph Neural Network
	Multi-hop reading comprehension	Sentence LSTM	Graph LSTM
	Relational reasoning	RN	MLP
		Recurrent RN	Recurrent Neural Network
		IN	Graph Neural Network
Image	Social Relationship Understanding	GRM	Gated Graph Neural Network
	Image classification	GCN	Graph Convolutional Network
		GGNN	Gated Graph Neural Network
		ADGPM	Graph Convolutional Network
		GSNN	Gated Graph Neural Network
	Visual Question Answering	GGNN	Gated Graph Neural Network
	Object Detection	RN	Graph Attention Network
	Interaction Detection	GPNN	Graph Neural Network
		Structural-RNN	Graph Neural Network
	Region Classification	GCNN	Graph CNN
	Semantic Segmentation	Graph LSTM	Graph LSTM
		GGNN	Gated Graph Neural Network
		DGCNN	Graph CNN
		3DGNN	Graph Neural Network
Science	Physics Systems	IN	Graph Neural Network
		VIN	Graph Neural Network
		GN	Graph Networks
	Molecular Fingerprints	NGF	Graph Convolutional Network
		GCN	Graph Convolutional Network
	Protein Interface Prediction	GCN	Graph Convolutional Network
	Side Effects Prediction	Decagon	Graph Convolutional Network
	Disease Classification	PPIN	Graph Convolutional Network
Knowledge Graph	KB Completion	IN	Graph Neural Network
	KG Alignment	GCN	Graph Convolutional Network
Combinatorial Optimization		structure2vec	Graph Convolutional Network
		GNN	Graph Neural Network
		GCN	Graph Convolutional Network
		AM	Graph Attention Network
Graph Generation		NetGAN	Long short-term memory
		GraphRNN	Rucurrent Neural Network
		Regularizing VAE	Variational Autoencoder
		GCPN	Graph Convolutional Network
		MolGAN	Relational-GCN

5 Definition of Chemoinformatics:

Chemoinformatics (alternatively known as cheminformatics) is a scientific discipline at the crossroads of information science, computer science, and chemistry. It is main focus on the information regarding interrelated chemical compounds to corresponding chemical and physical properties as well as descriptors. Computational Chemistry is the use of computer to perform calculations on molecules and generate the physical and even chemical properties of compounds.

There is more defenition of Chemoinformatics:

- Chemoinformatics is the use of computer and information-based methods to assist in transforming data into information or knowledge, and information or knowledge into decision, for solving chemical problems. [31]
- Chemoinformatics is the mixing of those information resources to transform data into information and information into knowledge for the intended purpose of making better decisions faster in the area of drug lead identification and optimization. [32]

The term chemoinformatics become today the favored term used for example by the Journal of chemoinformatics.

Mostly chemoinformatics is about chemical structures in one way or another.

The future challenges where chemoinformatics might play an important role are:

- Application of machine learning and artificial intelligence.
- Accelerating drug discovery and development processes.
- Chemoinformatics education and workforce development.
- To predicting Absorption, Distribution, Metabolism, Excretion, and Toxicity (ADMET) properties.
- Prediction of toxicity and environmental impact of chemicals.

6 Molecular Representations:

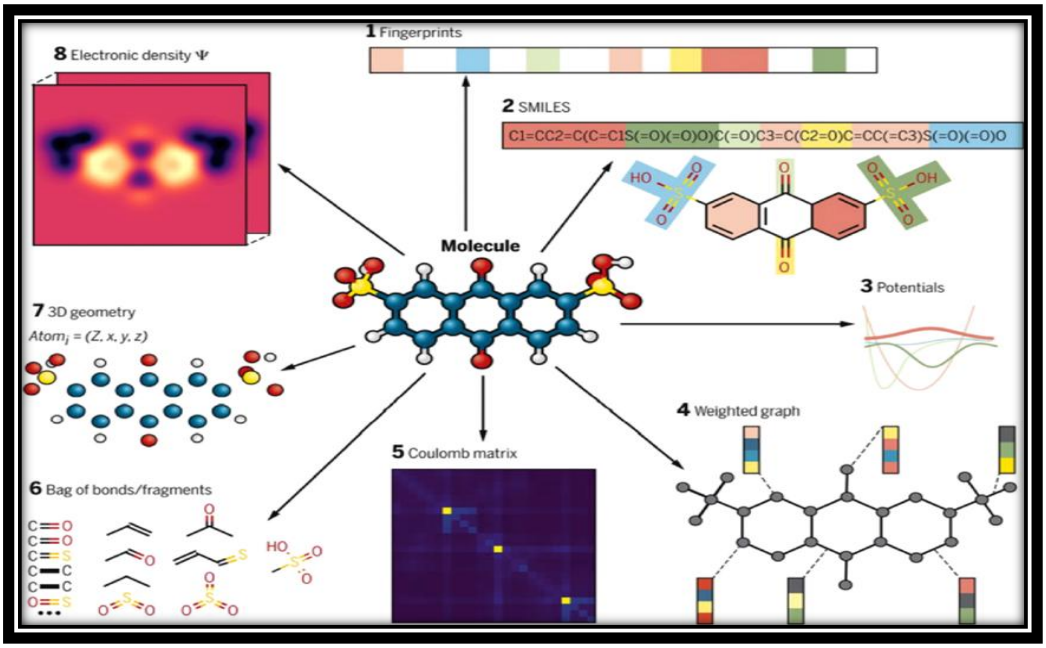


Figure 18: Different Types of Molecular Representations.

Molecular descriptors are mathematical representations of molecules that capture essential information about their structure and different aspects of their properties and characteristics. These descriptors play a crucial role in various scientific fields, including chemistry, pharmaceutical sciences.

The Molecular descriptors can be divided into 0D, 1D, 2D descriptors and 3D descriptors.

- **0D Descriptors:** 0D descriptors are simple molecular properties that do not depend on the structural representation of the molecule, such as molecular weight, sum of atomic properties, or counts of specific atom types. [33]

- **1D Descriptors:** 1D descriptors are derived solely from the molecular formula or simple properties, such as molecular weight, atom counts, and elemental composition, without considering the structural features or connectivity of the molecule. [33]

- **2D Descriptors:** 2D descriptors, also known as 2D fingerprints, are derived from the 2D structure of a molecule, taking into account the connectivity of atoms and the presence of specific substructures or molecular fragments. [34]

- **3D Descriptors:** 3D descriptors are derived from the 3D molecular structure, considering the spatial arrangement of atoms and the overall shape and volume of the molecule. These descriptors capture information about the electronic properties of molecules. [35]

- **Other 3D Descriptors**: Other 3D descriptors include more specific descriptors that capture various aspects of the 3D molecular structure, such as surface area, volume, shape descriptors, and energy-related descriptors (e.g., dipole moment, polarizability). [36]

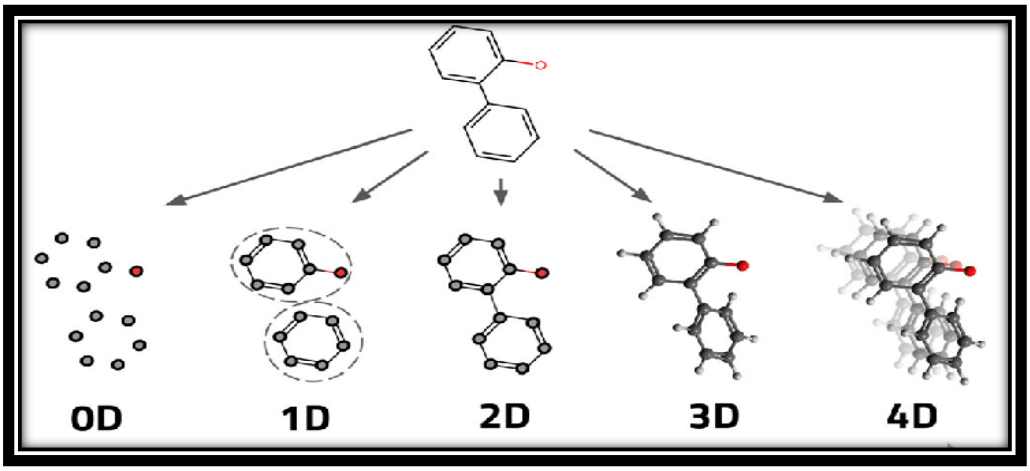


Figure 19: Level of Molecular Representations.

7 Molecular similarity measurement:

7.1 Introduction:

Molecular similarity is a pervasive concept in chemistry. It is essential to many aspects of chemical reasoning and analysis and is perhaps the fundamental assumption underlying medicinal chemistry. [37]

Therefore, the way molecular information is represented, called the representation problem, is of great importance for the type of molecular similarity analysis (MSA) that can be performed in a given situation. In this regard, sets, graphs, vectors, and functions are mathematical structures used to encode molecular information.

Similarity measures, also called similarity coefficients or indices, are functions that map pairs of compatible molecular representations (usually of the same mathematical form) to real numbers.

Determining the similarity of one "molecular object" to another is essentially a pattern matching exercise - often called a matching problem.

The result of the exercise is a value, the similarity measure that characterizes the identity, relatedness, proximity, similarity, or similarity of the molecular pair, as indicated by a "molecular pattern" consisting of a set of features.

7.2 Definition:

Molecular similarity measurement refers to the quantitative assessment of how structurally or functionally similar two or more molecules are to each other. This concept is crucial in various fields, including drug discovery, cheminformatics, and computational chemistry.

A concise definition could be:

Molecular similarity measurement is a computational method used to quantify the degree of resemblance between molecules based on their structural, physical, or chemical properties. [38]

7.3 Similarity Coefficients:

Similarity coefficients (also known as coefficients of association) are important measurement techniques used to quantify the extent to which objects resemble one another.

Similarity coefficients play an important role in many application domains. [39]

The types of similarity coefficients are available in the literature: association coefficients, correlation coefficients, distance coefficients, and probabilistic coefficients.

Table 3: The most commonly used similarity or distance coefficients for similarity searching in chemical database.

Coefficient	Formula for Continuous Variables	Binary Formula
Tanimoto (Jaccard)	$S_{AB} = \frac{\sum_{i=1}^{N} W_{iA} W_{iB}}{\sum_{i=1}^{N}(W_{iA})^2 + \sum_{i=1}^{N}(W_{iB})^2 - \sum_{i=1}^{N} W_{iA} W_{iB}}$ Range: -0.333 to +1	$S_{AB} = \frac{c}{a+b-c}$ Range: 0 to +1
Dice (Hodgkin)	$S_{AB} = \frac{2\sum_{i=1}^{N} W_{iA} W_{iB}}{\sum_{i=1}^{N}(W_{iA})^2 + \sum_{i=1}^{N}(W_{iB})^2}$ Range: -1 to +1	$\frac{2c}{a+b}$ Range: 0 to +1
Cosine (Carbó index)	$S_{AB} = \frac{\sum_{i=1}^{N} W_{iA} W_{iB}}{\sqrt{\sum_{i=1}^{N}(W_{iA})^2 \sum_{i=1}^{N}(W_{iB})^2}}$ Range: -1 to +1	$S_{AB} = \frac{c}{\sqrt{ab}}$ Range: 0 to +1
Euclidean distance	$D_{AB} = \sqrt{\sum_{i=1}^{N}(W_{iA} - W_{iB})^2}$ Range : 0 to ∞	$D_{AB} = \sqrt{a+b-2c}$ Range: 0 to N
Hamming (Manhattan) distance	$D_{AB} = \sum_{i=1}^{N} \lvert W_{iA} - W_{iB} \rvert$ Range: 0 to ∞	$D_{AB} = a+b-2c$ Range: 0 to N

Example:

A fingerprint is a binary sequence of Boolean values where 1 indicates a presence of a feature and a 0 indicates absence. Feature examples are ―200 < molecular weight <= 250‖ or ―aromatic ring count = 1‖. When comparing these fingerprints similarity scores are used. A much-applied method is the Tanimoto similarity.

Which for molecule A and B is defined as SAB= a / (a + b +c) where a bit are set common to both strings, b bits are set in the comparison string, c bit are set in the reference-structure string, and d bits are set in neither string.

Table 4: The Tanimoto Similarity Measures.

A	1	0	1	1	1	0	1	1	0	0	1	1	**b = 3**
B	0	0	1	1	0	0	1	0	1	0	1	1	**c = 1**

a = 5

$$S_{AB} = 5 / (5 + 3 + 1) = \mathbf{0.56}$$

7.4 Fingerprint similarity:

All of the unique fragments that exist in a molecule are hashed using a hashing technique to fit inside the length of the bit string in hashed fingerprints. Because it does not rely on a pre-existing list of structural components, this method allows for more generalizations. The nature of the chemical structures in the database, rather than the pieces in any specified list, characterizes the fingerprints created.

This technique indexes a set of patterns rather to utilizing a fragment dictionary. [40]

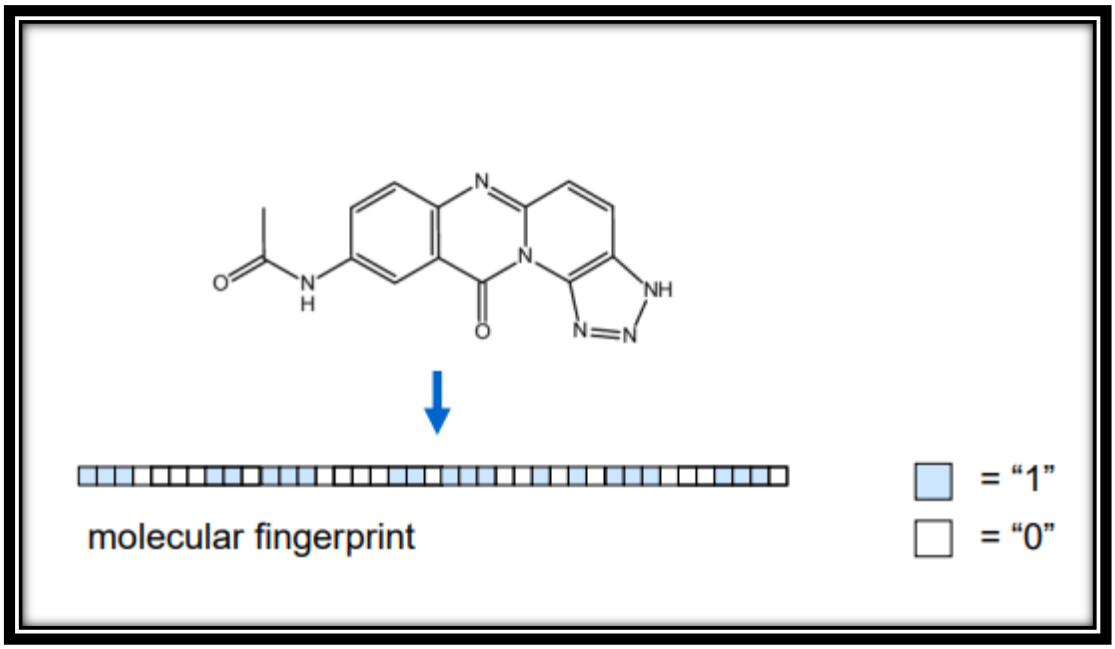


Figure 20: A molecular fingerprint representation

- Bit string representations of chemical structure and properties
- 2D and / or 3D features of a molecule are typically encoded as a vector of binary

values

- Keyed fingerprints:

- One bit position is associated with exactly one predefined feature.
- "1": presence of the feature in the molecule.
- "0": absence of the feature in the molecule Hashed fingerprints:
- One molecular feature is mapped to several bit positions applying a hash function
- It follows that a bit position is often set to "1" by more than one feature.

8 Quantitative structure- activity relationships:

8.1 Definition:

QSAR stands for Quantitative Structure-Activity Relationship. It's a mathematical modeling technique that connects the structural features of chemical substances to their biological activities or other properties. Essentially, QSAR aids in our comprehension of the relationship between structural modifications and a molecule's physical, chemical, and biological consequences. [41]

8.2 The Steps of QSAR study:

- The first major step in a QSAR study is the entry of the molecular structures and generation of the 3-D models. The 3-D molecular models are needed for geometric descriptor calculations.
- The second major step in a QSAR study is the generation of the molecular structure descriptors.
- The third step Selection of the most important descriptors and it can be achieved by using feature selection methods.
- The fourth major step in a QSAR study is the generation of the QSAR models using the descriptor sets.
- The fifth and last step is to validate the model by predicting the activity of compounds in the external prediction set. The results obtained by the predictions should be compared to those achieved for the training set and cross validation set to easily understand models fitness level. [42]

8.3 Advantages and disadvantages of QSAR:

8.3.1 Advantages:

- Able to predict activities of a large number of compounds with little to no prior experimental data on activity.
- Can reveal which molecular properties may be worth investigating further.
- Regarded as a “green chemistry” approach since chemical waste is not generated when performing in silico predictions.
- In vivo and in vitro experimentation can be very expensive and time-consuming. QSAR modeling reduces the need for testing on animals and/or on cell cultures and saves time. [43]

8.3.2 Disadvantages:

- Does not provide an in-depth insight on the mechanism of biological action.
- Some risk of highly inaccurate predictions of pharmacological or biological activity.[43]

9 Molecular property prediction:

9.1 Definition:

Molecular Property Prediction (MPP) aims to reveal the relationships between molecular structures and their properties. Scientists use MPP to screen molecules for various applications, including drug discovery and material design. [44]

9.2 Where we can use MPP:

9.2.1 Drug Discovery:

- Predicting the solubility, toxicity, and bioavailability of drugs in order to direct their design and optimization.
- Utilizing molecular characteristics to identify possible therapeutic candidates.
- Calculating a drug candidate's toxicity and off-target effects in order to spot possible safety concerns.

9.2.2 Materials Science:

- Predicting material properties for designing novel materials.

- Screening vast chemical regions to speed up the discovery of new compounds.
- Estimating a material's mechanical, thermal, optical, and electrical properties in order to create new materials with the appropriate characteristics**.**

9.2.3 Chemical Informatics:

- Quantitative Structure-Activity Relationship (QSAR) studies for understanding structure-property relationships.
- Estimating the properties of reaction and products

9.2.4 Environnemental Sciences :

- Predicting how chemical compounds will behave and behave in the environment,
- Evaluating the possible effects of chemicals on the environment, and finding safer substitutes.

9.3 Challenges Molecular Property Prediction:

Predicting molecular properties is a pivotal task in various scientific domains, including drug discovery, material science, and computational chemistry. However, it remains challenging due to data scarcity and class imbalance. [45]

10 Graph neural networks for molecular property prediction:

Molecular property prediction is a challenging task in drug discovery, and attracts increasingly more attention in the last decades. For example, designing molecular fingerprints based on the radial group of the molecular structure, and then uses the converted fingerprint for property prediction [46].

Specifically, the specific properties of a given molecule are determined through the application of a specific model.

However, traditional methods for predicting molecular properties often require:

- They require specialized experiments by chemists to validate the property names.
- They require high R&D costs and a lot of time.
- They require specialized models for different properties but lack generalization capabilities.

Graph Neural Networks (GNNs) are becoming more and more popular today because of their capacity to model graph structured data. Numerous fields, including social networks, knowledge graphs, and recommendation systems, have seen successes.

Another interesting usage for GNNs is the prediction of molecular properties, since molecules may be thought of as topological graphs with bonds acting as edges and atoms as nodes. The graph model of molecules may naturally capture the information of the molecular structure, including nodes (atoms) and edges (bonds), in contrast to alternative representations of molecules, as B. SMILES, which represent molecules as sequences but lose structural information.

This makes the task of predicting molecular properties similar to a supervised graph classification problem. [47]

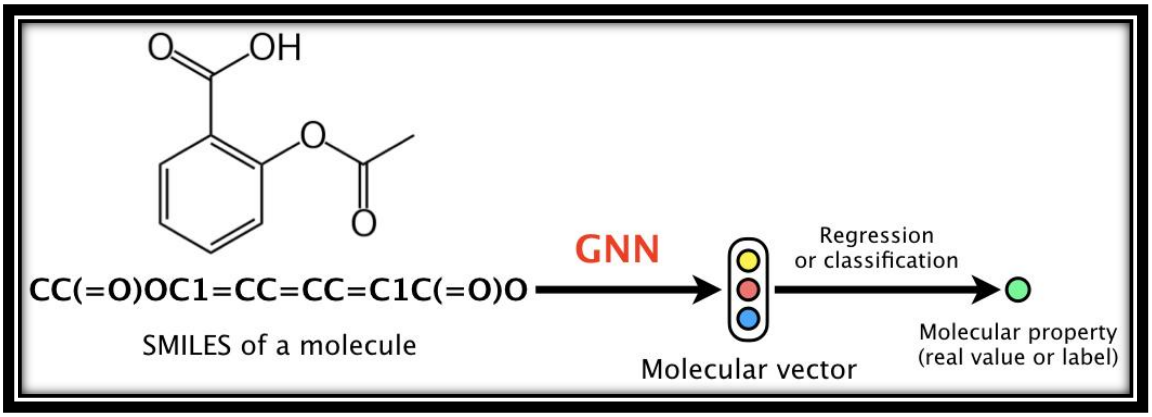


Figure 21: Graph neural networks for molecular property prediction

11 Conclusion:

Graph Theory, Graph Neural Networks (GNNs), and chemoinformatics constitute a formidable combination of fields that come together to improve our comprehension and use of complex data. With their adaptable node and edge structure, graphs offer a fundamental framework for describing a wide range of data kinds and relationships. Because they are made to handle graph-structured data, GNNs are highly effective at tasks like node, edge, and graph-level predictions. For this reason, they are widely used in domains including social network analysis, network security, and healthcare.

Chemoinformatics, which lies at the nexus of chemistry, computer science, and information science, uses computational techniques to solve problems related to chemistry, especially those related to drug discovery. To anticipate the biological actions of substances, it involves creating molecular descriptors and using quantitative structure-activity relationships (QSAR). The use of GNNs into chemoinformatics has significant promise for advancing scientific research and applications, despite obstacles such as class imbalance and scarce data.

To sum up, the dynamic and developing domains of graph theory, GNNs, and chemoinformatics together propel important advances in drug development, data analysis, and molecular property prediction, opening the door for more discoveries and improvements.

Chapter 3

Development Environment Programming tools

Chapter 3

1 Introduction: 40

2 Development environment: 40

2.1 Python: 40

2.2 Python module used: 41

2.3 Google Colaboratory: 43

2.3.1 Google Colaboratory definition: 44

2.4 Google Drive: 45

3 Summary 46

1 Introduction:

In this chapter, we present the details of the implementation of the system and the technical details, the hardware used the platform or the development environment, the programming language, and the different modules composing the software.

2 Development environment:

2.1 Python:



Is an interpreter, object-oriented, high-level programming language with dynamic semantics. Its high-level built in data structures, combined with dynamic typing and dynamic binding, make it very attractive for Rapid Application Development, as well as for use as a scripting. Python's simple, easy to learn syntax emphasizes readability and therefore reduces the cost of program maintenance. Python supports modules and packages, which encourages program modularity and code reuse. The Python interpreter and the extensive standard library are available in source or binary form without charge for all major platforms, and can be freely distributed. [48]

The language founded in the year 1991 by the developer Guido Van Rossum has the programming easy and fun to do.

2.1.1 Python Features:

Python provides lots of features that are listed below:

- **Easy to Learn and Use**: Python is easy to learn and use. It is developer-friendly and high level Programming language.
- **Expressive Language**: Python language is more expressive means that it is more understandable and readable.
- **Interpreted Language:** Python is an interpreted language i.e. interpreter executes the code line by line at a time. This makes debugging easy and thus suitable for beginners.

- **Cross-platform Language:** Python can run equally on different platforms such as Windows, Linux, UNIX and Macintosh etc. So, we can say that Python is a portable language.
- **Free and Open Source:** Python language is freely available at offical web address .The source code is also available. Therefore it is open source.
- **Object-Oriented Language:** Python supports object oriented language and concepts of classes and objects come into existence.
- **Extensible:** It implies that other languages such as C/C++ can be used to compile the code and thus it can be used further in our python code.
- **Large Standard Library:** Python has a large and broad library and provides rich set of module and functions for rapid application development.
- **GUI Programming Support**: Graphical user interfaces can be developed using Python.
- **Integrated:** It can be easily integrated with languages like C, C++ and JAVA etc…[49]

2.2 Python module used:

2.2.1 Numpy Module:



NumPy (Numerical Python) is a free and open source Python library used for working with arrays. It also has functions for working in domain of linear algebra, Fourier transform, and matrices. [50] Import the NumPy module:

- ✓ Import numpy as np.

2.2.2 Pandas Module:



Pandas is a fast, powerful, flexible and easy to use open source data analysis and manipulation tool, built on top of the Python programming language. [51]

Pandas module runs on top of NumPy and it is popularly used for data science and data analytics, and Import the pandas module:

- ✓ Import pandas as Pd.

2.2.3 RDKit Module:

2.2.4



RDKit is a widely used open-source toolkit for cheminformatics. One of its features is the conversion of molecules SMILES strings to 2D and 3D structures. The extension interface presents three tabs: Manage SMILES, Replace fragments, and Positional Analogue Scanning.

It was Developed and used at Rational Discovery for building predictive models for biological activity in 2006, and till the present the development continues to take hand for this tool. [52]

2.2.5 Tensrflow module:



TensorFlow is an open source software library for numerical computation using data flow graphs. The graph nodes represent mathematical operations, while the graph edges represent the multidimensional data arrays (tensors) that flow between them. This flexible architecture lets you deploy computation to one or more CPUs or GPUs in a desktop, server, or mobile device without rewriting code. TensorFlow also includes TensorBoard, a data visualization toolkit.

TensorFlow was originally developed by researchers and engineers working on the Google Brain team within Google's Machine Intelligence Research organization for the purposes of conducting machine learning and deep neural networks research. The system is general enough to be applicable in a wide variety of other domains, as well. [53]

2.2.6 MatplotlibModule :



Matplotlib is a low level graph plotting library in python that serves as a visualization utility. Matplotlib was created by John D. Hunter.

Matplotlib is open source and we can use it freely.

Matplotlib is mostly written in python; a few segments are written in C, Objective-C and Javascript for Platform compatibility [54] and Import the matplotlib module by:

- ✓ Import matplotlib.pyplot as plt.

2.2.7 Keras:



Keras is a Python-based open-source library of neural network components. Keras can run on top of TensorFlow, Theano, PlaidML, and other frameworks.

Although the library was designed to be modular and user-friendly, it began as part of a research project for the Open-ended Neuro-Electronic Intelligent Operating System, or ONEIROS. After Keras was released, it was not integrated into Google's TensorFlow core library until 2017. [55]

Import the Keras module:

- ✓ From tensorflow import keras.

2.3 Google Colaboratory:

2.3.1 Google Colaboratory definition:



Training a deep GNN requires powerful computing resources, thus for convenience we have used Google colaboratory. or “Colab” for short, is a product from Google Research. It allows anybody to write and execute arbitrary, python code through the browser, and is especially well suited to machine learning, data analysis and education. More technically, Colab is a hosted Jupyter notebook service (an open-source web application that allows creating and sharing documents that contain live code, equations, visualizations and narrative text) that requires no setup to use, while providing free access to computing resources including GPUs. Colab provides 12GB of RAM and Tesla K80 NVIDIA GPU. [56]

2.3.2 Benefits of Colab:

Besides being easy to use (which I’ll describe later), the Colab is fairly flexible in its configuration and does much of the heavy lifting for you.

- Python 2.7 and Python 3.6 support.
- Free GPU acceleration.
- Pre-installed libraries: All major Python libraries like TensorFlow, Scikit-learn, Matplotlib among many others are pre-installed and ready to be imported.
- Built on top of Jupyter Notebook.
- Collaboration feature (works with a team just like Google Docs): Google Colab allows developers to use and share Jupyter notebook among each other without having to download, install, or run anything other than a browser.
- Supports bash commands.
- Google Colab notebooks are stored on the drive. [57]

2.4 Google Drive:



Google Drive is a free service from Google that allows you to store files online and access them anywhere using the cloud**.** Google Drive also gives you access to free web-based applications for creating documents**,** spreadsheets**,** presentations**,** and more. [58]

Google Drive is a free cloud-based storage solution that enables users to:

- **Storing files:** By default, Google gives 15 GB of free storage space across Gmail, Google Drive and Google Photos. If you ever hit that limit, you can always pay for more space by upgrading to a Google One account.
- **Sharing files:** Files are private until you share them. But at any time, you can invite others to view, edit, or download files and documents.
- **Accessing files from anywhere:** Files can be accessed from any Smartphone, tablet or computer. Offline access is also available.
- **Comprehensive search:** Drive can recognize images and even text in scanned documents, making searching for files even easier.
- **Secure storage:** Drive files are protected by an AES256 or AES1283 encryption, the same security protocol used on other Google services.

Here is a breakdown of the most commonly used Google Drive applications:



Google Drive apps and functions

APP	DESCRIPTION	PRIMARY FUNCTIONS
Google Docs	Text documents	Proposals, reports, shared meeting notes
Google Sheets	Spreadsheets	Project plans, budget sheets
Google Slides	Presentations	Pitch decks, training modules, team presentations
Google Forms	Surveys	Customer satisfaction surveys, group polls
Google Drawings	Shapes, charts, and diagrams	Flowcharts, mind maps website wireframes
Google Sites	Websites	Team sites, project sites, resume sites

Figure 22: Google Drive applications and functions.

3 Summary:

In this chapter, we present the development environment, detail the different tools used the compilation in this work.

Chapter 4

Results and discussions

Chapter 4

1 Introduction:.....47

2 Explanation of our approach:.....47

2.1 Dataset:.....47

2.2 Data download:.....47

3 Data preprocessing:.....49

3.1 Splitting the data into stratified train and test sets:.....49

4 Develop the Model.....49

4.1 GCN model:.....49

4.2 Evaluation:.....51

5 Result model.....53

5.1 The Result of Property.....53

5.2 Table of Result model:.....58

5.3 Comparison between the established model and the piper model:.....60

5.3.1 THE Observation.....60

5.3.2 The conclusion.....61

6 Summary.....61

1 Introduction:

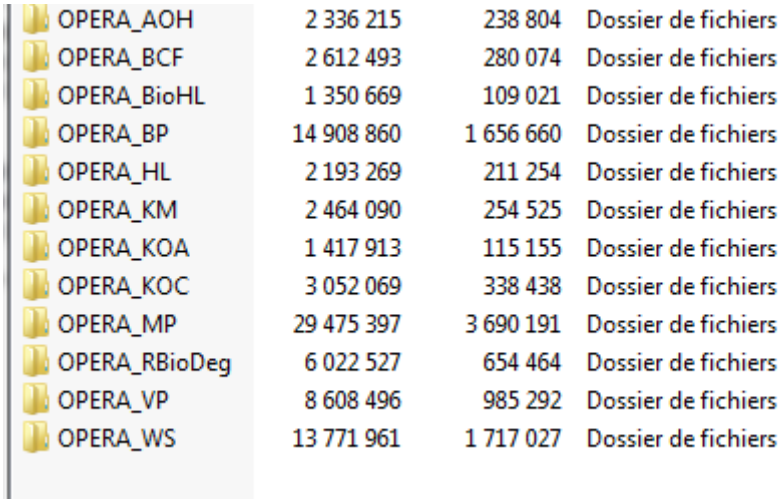
We present in this chapter the implementation of our proposed approach for property, start by describing the data set, advance data processing, training and validation of the model. Then we explain the steps of our implementation, in the end we interpret and comment the results Obtained.

2 Explanation of our approach:

2.1 Dataset:

We used the datasets OPERA Models from the publicly available PHYSPROP database consisting of a set of 12 common physicochemical and environmental fate properties.

2.2 Data download:



OPERA_AOH	2 336 215	238 804	Dossier de fichiers
OPERA_BCF	2 612 493	280 074	Dossier de fichiers
OPERA_BioHL	1 350 669	109 021	Dossier de fichiers
OPERA_BP	14 908 860	1 656 660	Dossier de fichiers
OPERA_HL	2 193 269	211 254	Dossier de fichiers
OPERA_KM	2 464 090	254 525	Dossier de fichiers
OPERA_KOA	1 417 913	115 155	Dossier de fichiers
OPERA_KOC	3 052 069	338 438	Dossier de fichiers
OPERA_MP	29 475 397	3 690 191	Dossier de fichiers
OPERA_RBioDeg	6 022 527	654 464	Dossier de fichiers
OPERA_VP	8 608 496	985 292	Dossier de fichiers
OPERA_WS	13 771 961	1 717 027	Dossier de fichiers

Figure 23: The dataset used.

- **Example of one Molecule in the dataset** :

```
100002
  -INDIGO-08031722252D

  5  4  0  0  0  0  0  0  0  0999 V2000
   -0.8000   -1.3856    0.0000 Cl  0  0  0  0  0  0  0  0  0  0  0  0
    0.0000    0.0000    0.0000 C   0  0  0  0  0  0  0  0  0  0  0  0
    0.8000    1.3856    0.0000 Cl  0  0  0  0  0  0  0  0  0  0  0  0
   -1.3856    0.8000    0.0000 Cl  0  0  0  0  0  0  0  0  0  0  0  0
    1.3856   -0.8000    0.0000 Cl  0  0  0  0  0  0  0  0  0  0  0  0
  1  2  1  0  0  0  0
  2  3  1  0  0  0  0
  2  4  1  0  0  0  0
  2  5  1  0  0  0  0
M  END
> <ChemID>
100002

> <source_casrn>
56-23-5

> <CAS>
56-23-5

> <SMILES>
ClC(Cl)(Cl)Cl

> <NAME>
Tetrachloromethane

> <preferred_name>
Carbon tetrachloride

> <iupac>
tetrachloromethane

> <MPID>
18816

> <dsstox_substance_id>
DTXSID8020250

> <Canonical_QSARr>
ClC(Cl)(Cl)Cl

> <InChI_Code_QSARr>
InChI=1S/CCl4/c2-1(3,4)5

> <InChI Key_QSARr>
VZGDMQKNWNREIO-UHFFFAOYSA-N

> <LogKOA>
2.79

$$$$
```

Figure 24: Example of one molecule from the Dataset.

3 Data preprocessing:

3.1 Splitting the data into stratified train and test sets:

The full dataset was randomly divided into two parts, training set and test sets for any propriety from the data.

The optimal models were built on randomly selected training sets (75%) and test sets (25%).

Table 5: The number of training and testing dataset for each property.

Property	Training (75%) Dataset	Test (25%) Dataset
AOH	516	176
BCF	469	157
BioHL	112	38
BP	4077	1358
HL	441	150
KM	405	136
KOA	202	68
KOC	545	184
LogP	10,537	3513
MP	6486	2167
VP	2034	679
WS	3158	1066

4 Develop the Model

4.1 GCN model:

The model shown is a network model that uses different layers to process data.

The correlated data is analyzed by the model by processing its initial set of inputs through multiple Graph-Based Convolutional (GCN) layers. Important features are extracted by passing the outputs through multiple GR layers that seem to be working on the same outputs. The GR layers' outputs are all combined into one sizable layer. The final output is produced in the desired shape by progressively reducing the dimensions through the application of multiple dense layers.

Table 6 : The structure of the model.

```
Model: "model"
__________________________________________________________________________________________________
 Layer (type)                Output Shape              Param #   Connected to
==================================================================================================
 input_1 (InputLayer)        [(None, None, 5)]         0         []

 input_2 (InputLayer)        [(None, None, None)]      0         []

 gcn_layer (GCNLayer)        ((None, None, 32),        160       ['input_1[0][0]',
                              (None, None, None))                  'input_2[0][0]']

 gcn_layer_1 (GCNLayer)      ((None, None, 32),        1024      ['gcn_layer[0][0]',
                              (None, None, None))                  'gcn_layer[0][1]']

 gcn_layer_2 (GCNLayer)      ((None, None, 32),        1024      ['gcn_layer_1[0][0]',
                              (None, None, None))                  'gcn_layer_1[0][1]']

 GRLayer (GRLayer)           (None, 32)                0         ['gcn_layer_2[0][0]',
                                                                  'gcn_layer_2[0][1]']

 GRLayer2 (GRLayer2)         (None, 32)                0         ['gcn_layer_2[0][0]',
                                                                  'gcn_layer_2[0][1]']

 GRLayer7 (GRLayer7)         (None, 32)                0         ['gcn_layer_2[0][0]',
                                                                  'gcn_layer_2[0][1]']

 GRLayer8 (GRLayer8)         (None, 32)                0         ['gcn_layer_2[0][0]',
                                                                  'gcn_layer_2[0][1]']

 concatenate (Concatenate)   (None, 128)               0         ['GRLayer[0][0]',
                                                                  'GRLayer2[0][0]',
                                                                  'GRLayer7[0][0]',
                                                                  'GRLayer8[0][0]']

 dense (Dense)               (None, 128)               16512     ['concatenate[0][0]']

 dense_1 (Dense)             (None, 64)                8256      ['dense[0][0]']

 dense_2 (Dense)             (None, 64)                4160      ['dense_1[0][0]']

 dense_3 (Dense)             (None, 1)                 65        ['dense_2[0][0]']

==================================================================================================
Total params: 31201 (121.88 KB)
Trainable params: 31201 (121.88 KB)
Non-trainable params: 0 (0.00 Byte)
__________________________________________________________________________________________________
```

We also create a plot to visualize the structure of the network:



Figure 25: Plot of Deep Learning Model**.**

4.2 Evaluation:

The following graph shows the error rate of the model relative to the number of times it is used for some properties:

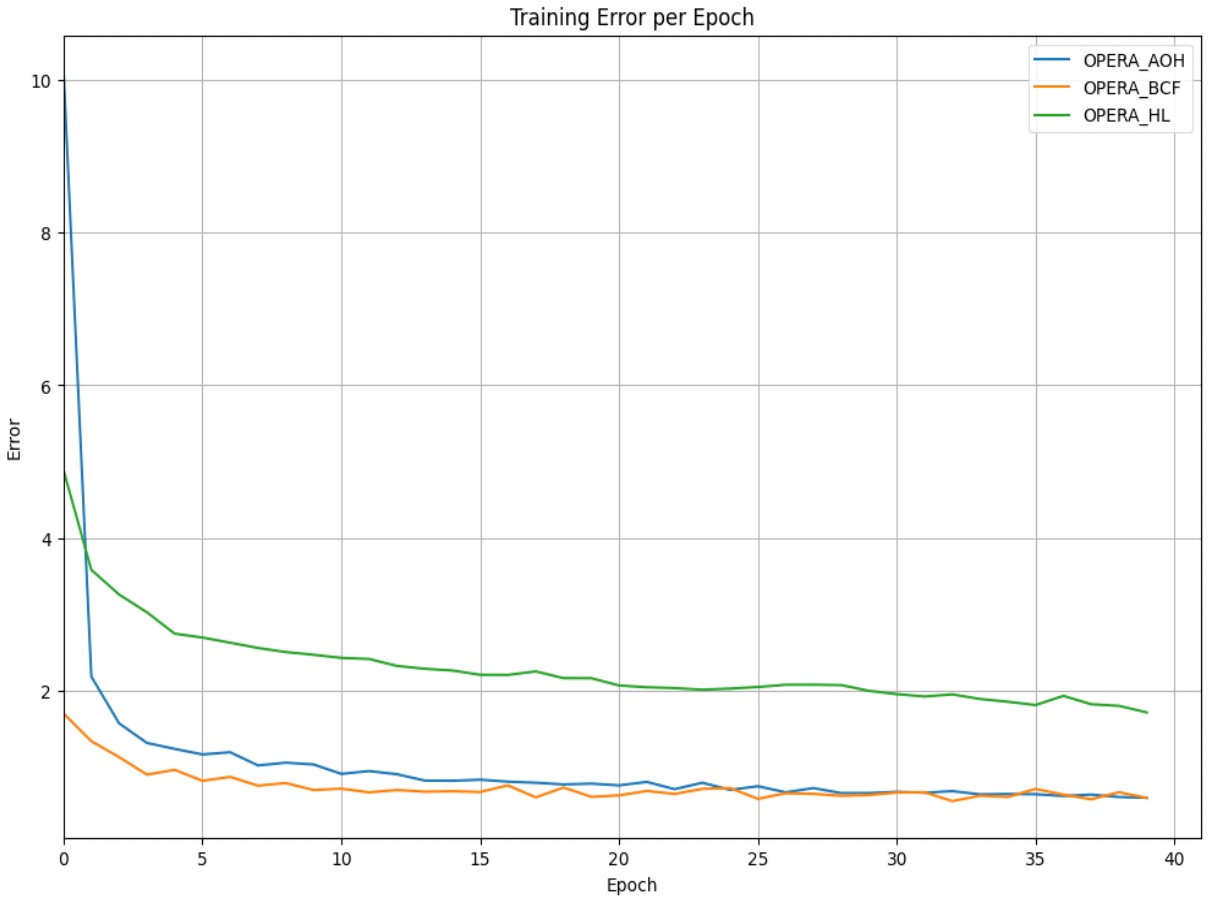


Figure 26 : The error rate of the model.

These values are usually used to evaluate the performance of the model during training, as developers seek to reduce the value of the error or loss with each passing episode of the training cycles. It appears that the training error or loss function (Loss function) during those cycles decreases, indicating a clear improvement in the performance of the model during the training stages across the 40 epochs.

5 Result model

5.1 The Result of Property

- **Property Log_KOA**

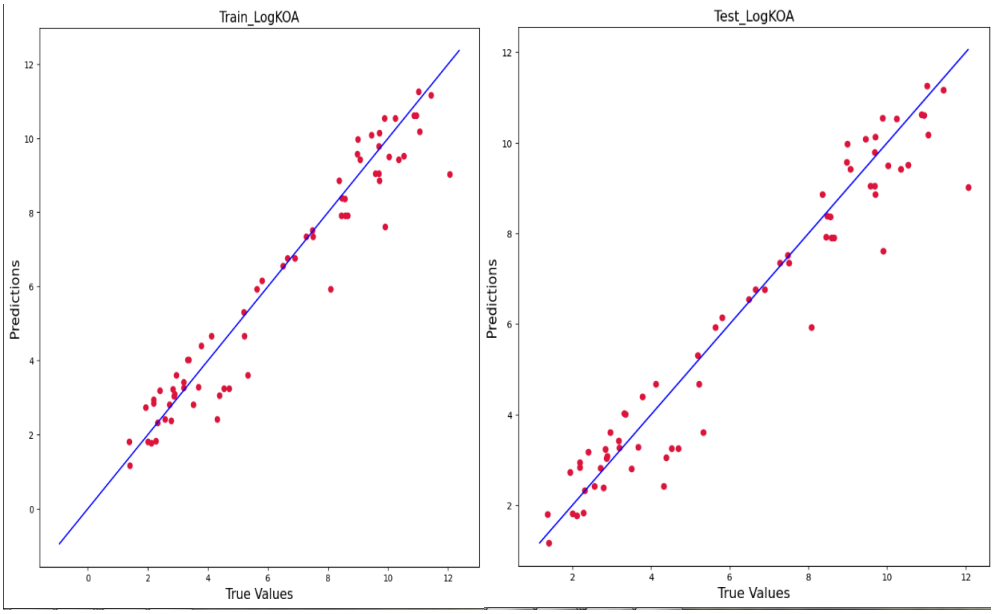


Figure 27 : Train logKOA **Figure 28 :** Test LogKOA

- **Property Log_VP**

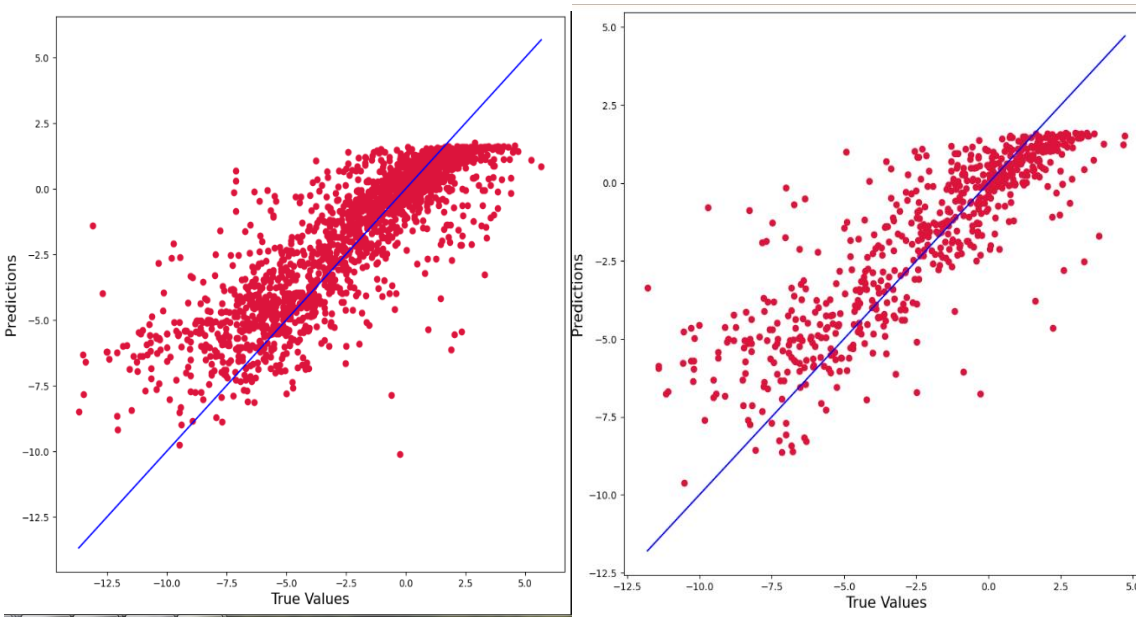


Figure 29:Train LogVP **Figure** 30: Test LogVP

- **Property WS**

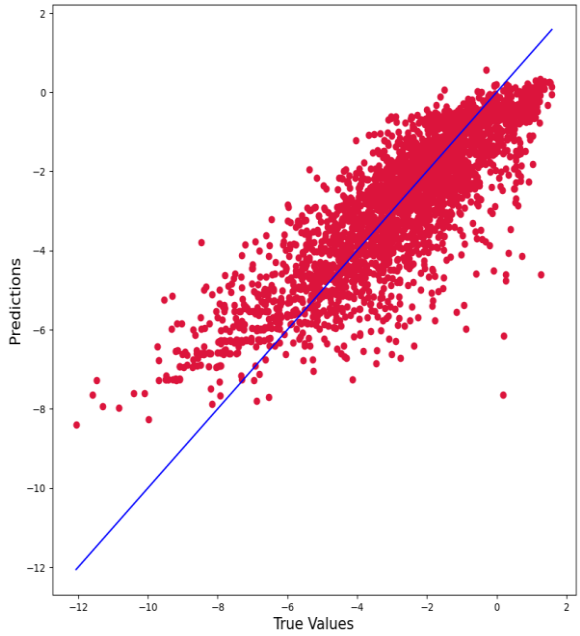


Figure 31: Train LogMoler

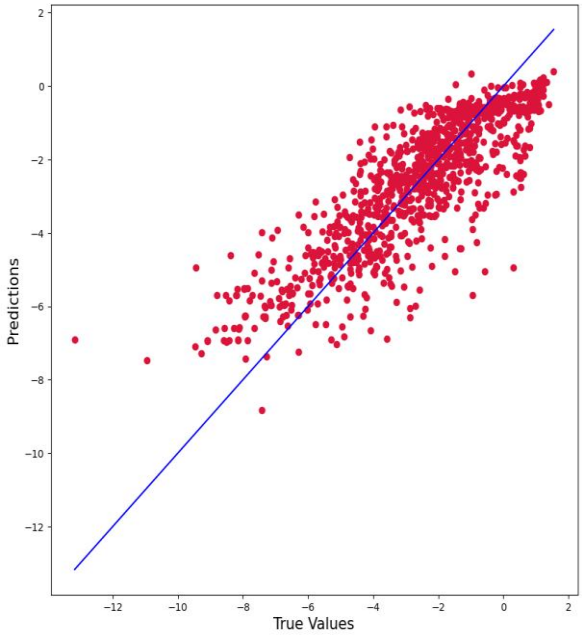


Figure 32: Test LogMoler

- **Property AOH**

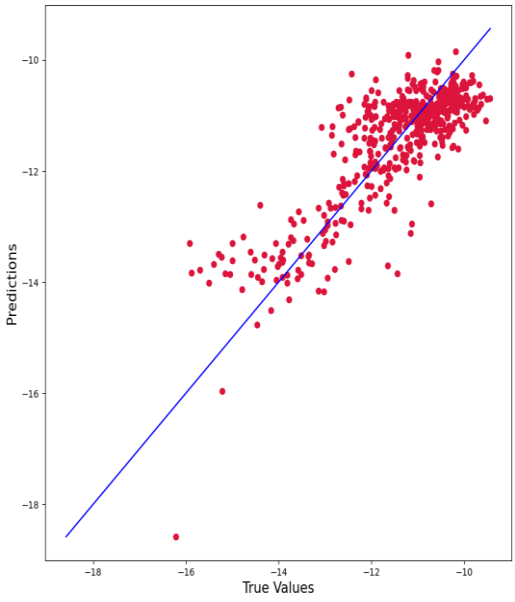


Figure 33:train logOH

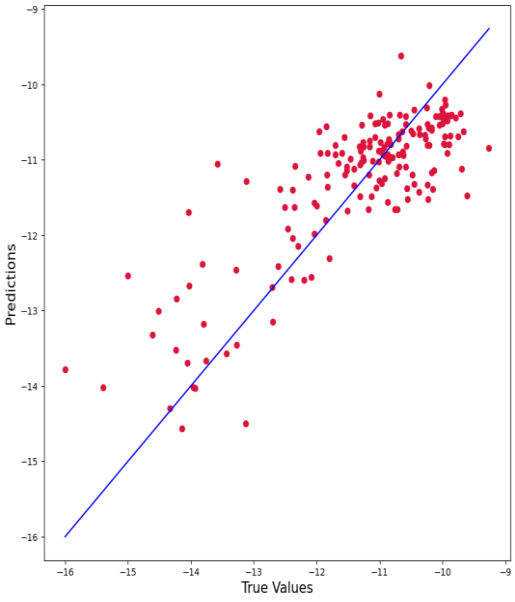


Figure 34: Test logOH

- **Property Log_BCF**

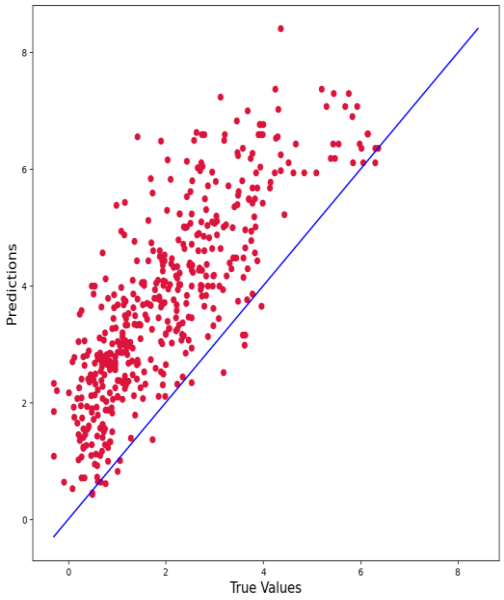


Figure 35: Train LogBCF

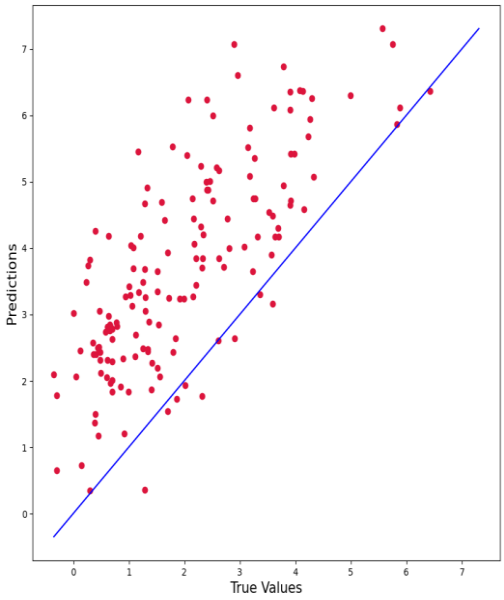


Figure 36: Test LogBCF

- **Property BP**

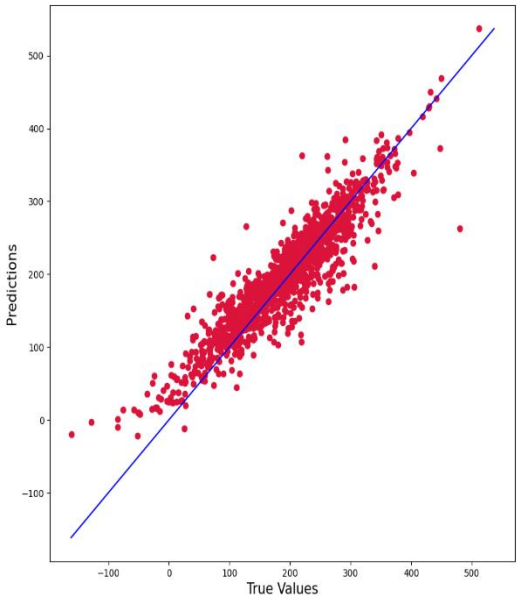


Figure 37: Train BP.

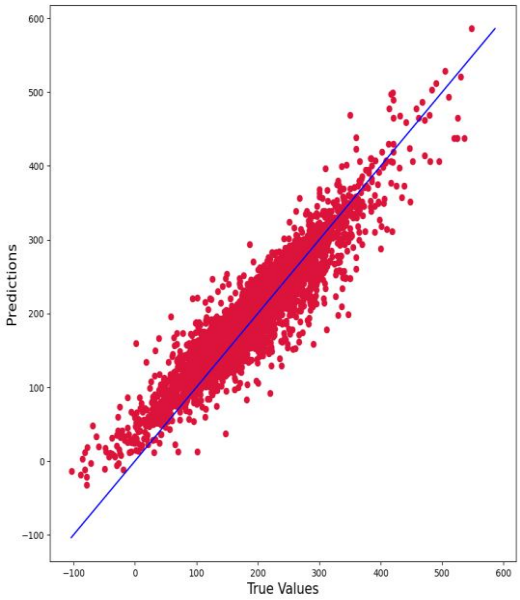


Figure 38: Test BP.

- **Property Log_HL**

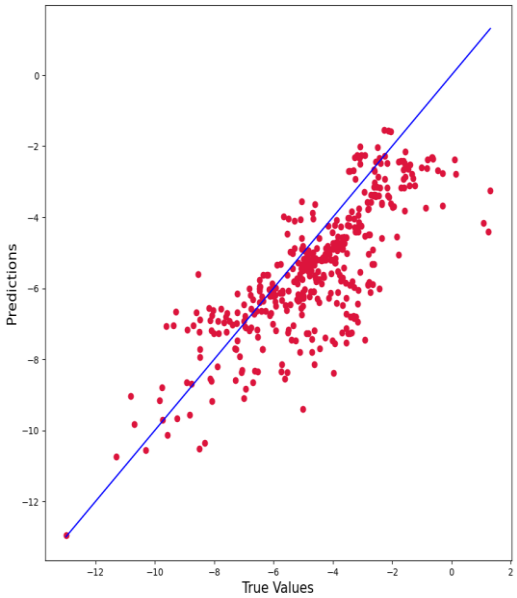


Figure 39: Train LogHL

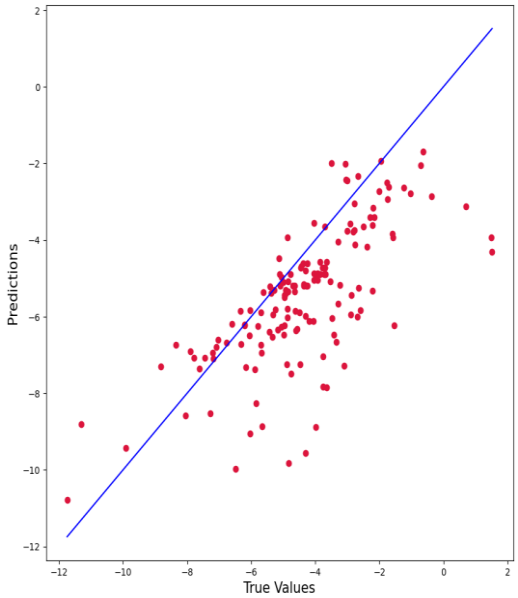


Figure 40: Test LogHL.

- **Property Log_KM**

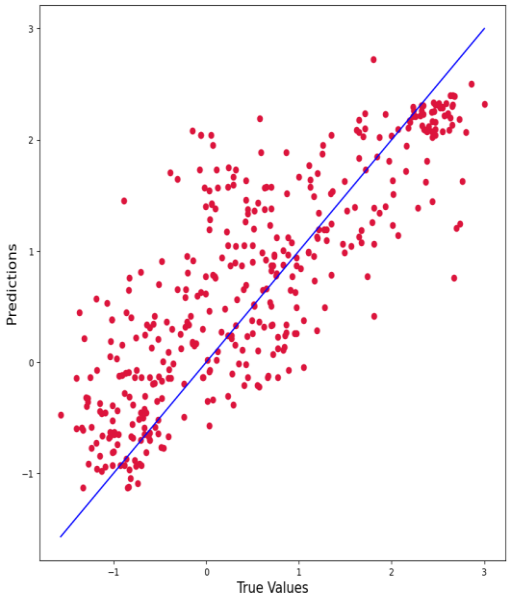


Figure 41: train LogKmHL

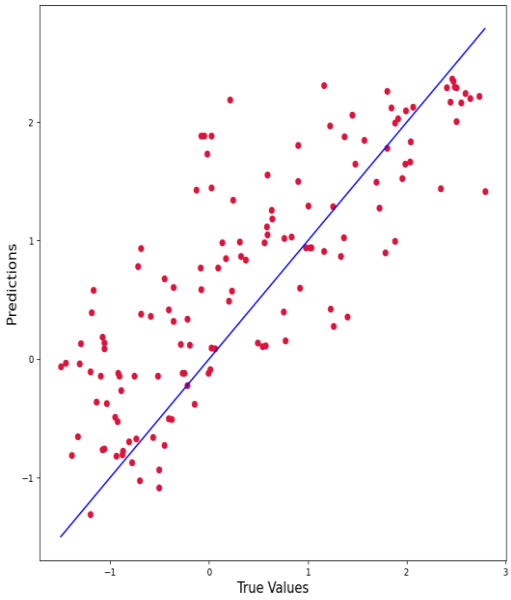


Figure 42: test LogKmHL

- **Property MP**

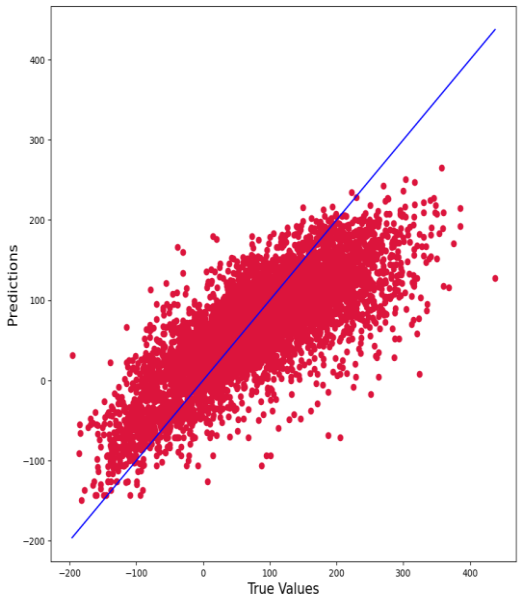


Figure 43: Train MP

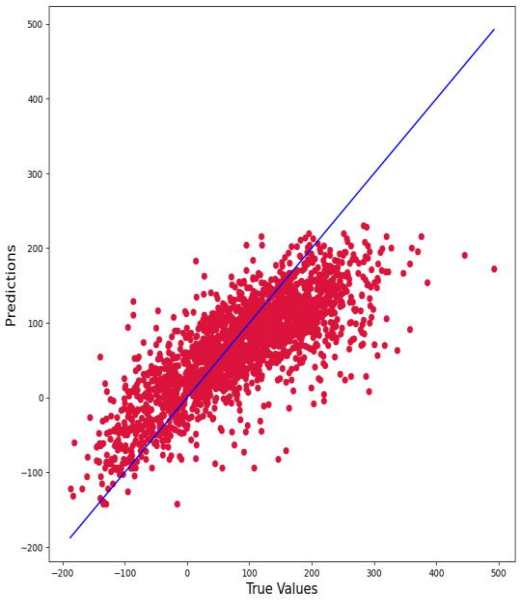


Figure 44: Test MP

- **Property Log_KOC**

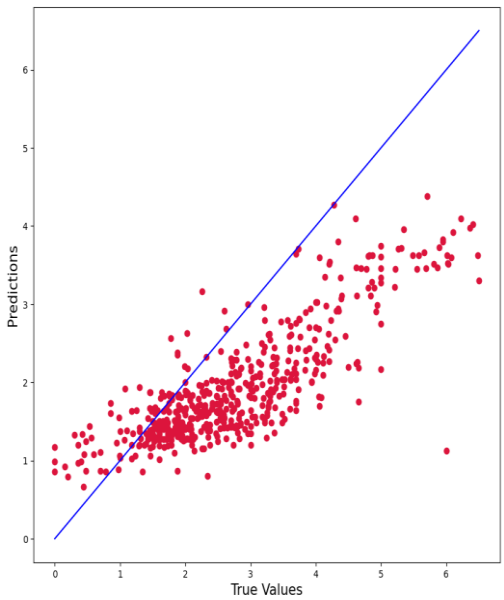


Figure 45 : Train LogKOC

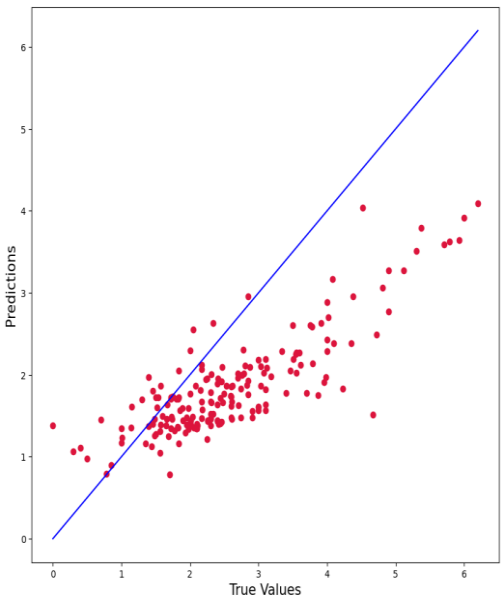


Figure 46:Test Log KOC

➢ **Property Log_BioHL**

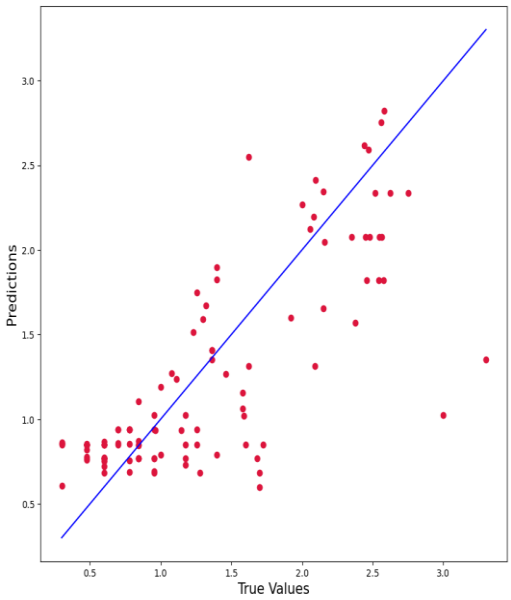


Figure 47: Train LogHalfLife

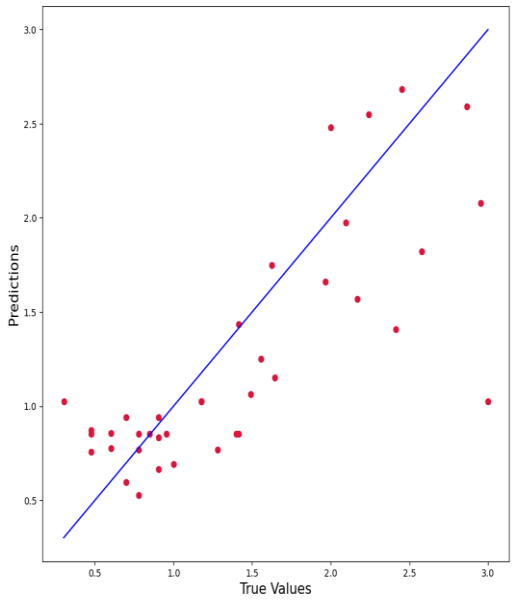


Figure 48: Test LogHalfLife

5.2 Table of Result model:

Results of common performance metrics (MSE, RMSE, MAE, R2) in evaluating the performance of a prediction model on training data and test data.

Table 7: Result the Established model**.**

Established model	MSE		RMSE		MAE		R2	
	train	test	train	Test	train	test	train	Test
LogOH	0.50	0.56	0.71	0.75	0.54	0.57	0.68	0.67
LogBCF	3.91	4.11	1.97	2.02	1.74	1.77	-0.96	-1.01
LogHalfLife	0,21	0.26	0.46	0.51	0.34	0.36	0.59	0.54
BP	971.29	1089.08	31.16	33.00	23.3	24.3	0.86	0.84

LogHL	2.40	3.42	1.55	1.85	1.20	1.36	0.45	0.18
LogKmHL	0.52	0.59	0.72	0.77	0.55	0.59	0.62	0.57
LogKOA	0.61	0.73	0.78	0.85	0.57	0.60	0.93	0.92
LogKOC	1.25	1.06	1.11	1.03	0.90	0.84	0.20	0.18
MP	3872.15	4445.78	62.22	66.67	47.13	50.11	0.60	0.57
LogVP	2.02	2.25	1.42	1.50	1.00	1.06	0.84	0.82
LogMolar	1.41	1.48	1.18	1.21	0.90	0.93	0.72	0.72

- **Mean squared error (MSE):** the average squared difference between the value observed in a statistical study and the values predicted from a model.
- **The root mean square error (RMSE)** measures the average difference between a statistical model's predicted values and the actual values.
- MAE **(Mean Absolute Error)**: Measures the average of the absolute errors
- R^2 **(Coefficient of Determination)**: Represents the proportion of the variance for the dependent variable that's explained by the independent variables in the model.

We have give a table with a list of several regression models and their test and training dataset performance characteristics, distinct metrics, such as Mean Squared Error (MSE), Root Mean Squared Error (RMSE), Mean Absolute Error (MAE), and Coefficient of Determination (R2), are represented by the columns, each of which indicates a distinct model.

Explaining the Results of Each Model:

- **LogOH :**

  On both training and testing datasets, this model consistently performs well; high R2 values signify strong model performance.

- **LogKmHL:**
  Across training and testing datasets, this model exhibits strong generalization and reasonably consistent performance measures.
- **LogKOA:**
  High R2 values suggest that this model performs exceptionally well and accounts for the majority of the variance. The testing and training datasets exhibit consistent performance.
- **LogKOC:**
  The low R2 values imply that only a small portion of the data variation is explained by this model.
- **LogMolar:**
  High R2 values and consistent error metrics between training and testing datasets indicate the effectiveness of this model.

In brief:

Best Performing Models: LogKOA, BP, and LogVP, their performance indicators are consistent and their R2 values are high.

Generalization: The majority of models perform consistently across training and testing datasets, allowing for good generalization.

5.3 Comparison between the established model and the piper model:

Table 8: The result of piper model: [59]

PIPER MODEL	**MSE**		**RMSE**		**R2**	
	Train	**test**	**Train**	**test**	**train**	**Test**
LogOH	1.25	1.51	1.12	1.23	0.85	0.83
LogBCF	0.28	0.40	0.53	0.64	0.85	0.83
LogHalfLife	0.06	0.14	0.26	0.38	0.88	0.75

BP	486.64	487.52	22.06	22.08	0.93	0.93
LogHL	3.64	3.31	1.91	1.82	0.84	0.85
LogKmHL	0.25	0.38	0.5	0.62	0.82	0.73
LogKOA	0.42	0.46	0.65	0.68	0.95	0.96
LogKOC	0.29	0.37	0.54	0.61	0.81	0.71
MP	2412.77	2732.15	49.12	52.27	0.75	0.74
LogVP	1.16	1	1.08	1	0.91	0.92
LogMolar	0.67	0.73	0.82	0.86	0.87	0.86

5.3.1 The Observation:

- **LogOH:** For both the train and test sets, the established MODEL has lower MSE and RMSE, but PIPER has higher R2 values. On this measure, Established MODEL does better overall.
- **LogBCF:** The PIPER MODEL performs much better across the board than the Established MODEL.
- **LogHalfLife:** The PIPER MODEL performs better than the Established MODEL throughout the board.
- **BP:** In every statistic, the PIPER MODEL performs better than the Established MODEL.
- **LogHL:** While the Established MODEL has a lower train MSE, the PIPER MODEL has a better R2 and a lower test MSE. In general, PIPER is superior.
- **LogKmHL:** In every metric, the PIPER MODEL performs better than the Established MODEL.
- **LogKOA:** The PIPER MODEL performs better than the Established MODEL throughout the board.

- **LogKOC:** The PIPER MODEL performs better than the Established MODEL throughout the board.

5.3.2 The conclusion:

When compared to the Established MODEL, the PIPER MODEL performs better overall across the majority of features and metrics, suggesting that it is the superior model.

6 Summary:

In this chapter, we introduce the implementation, tests and results obtained from our proposed approach, about the model as training, assembling, testing the model and obtaining the results.

General conclusion:

The convergence of cheminformatics, deep learning, and graph theory has greatly boosted scientific investigations and applications. Graph Neural Networks (GNNs) are an efficient way to handle graphs, which are necessary for modeling complicated relationships. GNNs are particularly good at tasks like link prediction and node classification.

Because deep learning, a branch of machine learning, can learn from and get better at analyzing large volumes of data, it has completely changed the way that data is processed. This feature has improved a number of applications, including natural language processing and picture and speech recognition.

Cheminformatics, a field that combines information science, computer science, and chemistry, is concerned with organizing and analyzing chemical data. Methods such as molecular similarity assessments and Quantitative Structure-Activity Relationships (QSAR) are essential for predicting chemical qualities, expediting drug discovery, and decreasing the requirement for lengthy experimental testing.

Researchers can create more precise models for predicting chemical properties, increasing the effectiveness of drug discovery and development, by combining GNNs and deep learning with cheminformatics.

References

[1] Nguyen, D. "Some Mathematical Perspectives of Graph Neural Networks." 2022, page 3.

[2] Watson, D. "Graph Neural Networks: Definition, Types, Applications." 28 September 2023.

[3] Ravikirqn, A. S. "What is Graph in Data Structure & Types of Graph?" 26 July 2023.

[4] Ma, Y., and Tang, J. "Deep Learning on Graphs."

[5] GeeksforGeeks. "Graph and Its Representations." 2024. https://www.geeksforgeeks.org/graph-and-its-representations/

[6] Liu, Z., and Zhou, J. "Introduction to Graph Neural Networks." 2020, page 11.

[7] Shruti, M. "AI vs Machine Learning vs Deep Learning: Know the Differences." https://www.simplilearn.com/tutorials/artificial-intelligence-tutorial/ai-vs-machine-learning-vs-deep-learning#what_is_artificial_intelligence

[8] Betz, S. "7 Types of Artificial Intelligence." https://builtin.com/artificial-intelligence/types-of-artificial-intelligence

[9] Coursera. "What is Machine Learning?" https://www.coursera.org/articles/what-is-machine-learning

[10] GeeksforGeeks. "Types of Machine Learning." https://www.geeksforgeeks.org/types-of-machine-learning

[11] IBM. "Unsupervised Learning." https://www.ibm.com/topics/unsupervised-learning

[12] IBM. "Semi-Supervised Learning." https://www.ibm.com/topics/semi-supervised-learning

[13] AskAnyDifference. "Difference Between Semi-Supervised and Reinforcement Learning." https://askanydifference.com/difference-between-semi-supervised-and-reinforcement-learning/

[14] Google Cloud. "What is Deep Learning." https://cloud.google.com/discover/what-is-deep-learning

[15] Coursera Staff. "What Is Deep Learning? Definition, Examples, and Careers." 26 March 2024. https://www.coursera.org/articles/what-is-deep-learning

[16] Yamashita, R. "Convolutional Neural Networks: An Overview and Application in Radiology." 22 June 2018.

[17] Mahmoudi, M. A. "Deep Learning for Emotion Recognition." Doctorate Thesis, Mustapha Stambouli University Mascara, 07 July 2022, pages 7-8.

[18] GeeksforGeeks. "Introduction to Convolution Neural Network." 14 March 2024. https://www.geeksforgeeks.org/introduction-convolution-neural-network/

[19] Engati. "Convolutional Neural Network." https://www.engati.com/glossary/convolutional-neural-network

[20] Menzli, A. "Graph Neural Network and Some of GNN Applications: Everything You Need to Know." 11 September 2023.

[21] Kipf, T. N., & Welling, M. "Semi-Supervised Classification with Graph Convolutional Networks." 2017.

[22] Khemani, B., Patil, S., Kotecha, K., & Tanwar, S. "A Review of Graph Neural Networks: Concepts, Architectures, Techniques, Challenges, Datasets, Applications, and Future Directions." 16 January 2024.

[23] Veličković, P., Cucurull, G., Casanova, A., Romero, A., Lio, P., & Bengio, Y. "Graph Attention Networks." 2018.

[24] Hamilton, W., Ying, Z., & Leskovec, J. "Inductive Representation Learning on Large Graphs." 2017.

[25] Palle, K., Pires, B. A., & Ribeiro, B. "Graph Recurrent Neural Networks." 2020.

[26] DataCamp. "A Comprehensive Introduction to Graph Neural Networks (GNNs)." July 2022. https://www.datacamp.com/tutorial/comprehensive-introduction-graph-neural-networks-gnns-tutorial

[27] Machine Learning Knowledge. "Graph Neural Networks (GNN) Explained for Beginners." 18 April 2023. https://machinelearningknowledge.ai/graph-neural-networks-gnn-explained-for-beginners/

[28] Amirian, S. "Challenges of Leveraging Graphs in Machine Learning." 6 October 2023.

[29] SEON. "Graph Neural Network (GNN)." https://seon.io/resources/dictionary/graph-neural-network-gnn/

[30] Sharma, P. "What are Graph Neural Networks, and How Do They Work?" 05 December 2023.

[31] Leach, A. R., & Gillet, V. J. "An Introduction to Chemoinformatics." 2007.

[32] Brown, F. K. "Chemoinformatics: What is it and How Does it Impact Drug Discovery?" Annual Reports in Computational Chemistry, 1, 31-32. https://doi.org/10.1016/S1574-1400(05)01002-5. 2005.

[33] Todeschini, R., & Consonni, V. "Molecular Descriptors for Chemoinformatics." Vol. 41, pages 1-1012. Wiley-VCH, Weinheim, Germany. 2009.

[34] Rogers, D., & Hahn, M. "Extended-Connectivity Fingerprints." Journal of Chemical Information and Modeling, 50(5), 742-754. 2010.

[35] Gasteiger, J., & Marsili, M. "Iterative Partial Equalization of Orbital Electronegativity—a Rapid Access to Atomic Charges." Tetrahedron, 36(22), 3219-3228. 1980.

[36] Sauer, W. H., & Schwarz, M. K. "Molecular Shape Diversity of Combinatorial Libraries: A Prerequisite for Broad Bioactivity." Journal of Chemical Information and Computer Sciences, 43(3), 987-1003. 2003.

[37] Maggiora, G. M., & Shanmugasundaram, V. "Molecular Similarity Measures." https://doi.org/10.1385/1-59259-802-1:001.

[38] Willett, P. "Similarity-based Virtual Screening Using 2D Fingerprints." *Drug Discovery Today*, 11(23-24), 1046-1053. 2006.

[39] Wong, K. S., & Kim, M. H. "Privacy-Preserving Similarity Coefficients for Binary Data." https://doi.org/10.1016/j.camwa.2012.02.028.

[40] Cereto-Massagué, A., Ojeda, M. J., Valls, C., Mulero, M., Garcia-Vallvé, S., & Pujadas, G. "Molecular Fingerprint Similarity Search in Virtual Screening." Methods, vol. 71, no. C, pp. 58–63, 2015. https://doi.org/10.1016/j.ymeth.2014.08.005.

[41] ECHA. "Practical Guide: How to Use and Report QSARs." Version 3.1 – July 2016. https://echa.europa.eu/documents/10162/13655/pg_report_qsars_en.pdf/407dff11-aa4a-4eef-a1ce-9300f8460099

[42] Sahigara, F., Mansouri, K., Ballabio, D., Mauri, A., Consonni, V., & Todeschini, R. "Comparison of Different Approaches to Define the Applicability Domain of QSAR Models." Molecules, 17, 4791-4810. 2012.

[43] PharmaFactz. "Medicinal Chemistry - Introduction to Quantitative Structure-Activity Relationships (QSAR)." https://pharmafactz.com/medicinal-chemistry-introduction-to-quantitative-structure-activity-relationships-qsar/

[44] Wang, Z., Lin, Z., Lin, W., Yang, M., Zeng, M., & Tan, K. C. "Explainable Molecular Property Prediction: Aligning Chemical Concepts with Predictions via Language Models."

[45] Zhang, R., Wu, C., Yang, Q., Liu, C., Wang, Y., Li, K., Huang, L., & Zhou, F. "MolFeSCue: Enhancing Molecular Property Prediction in Data-Limited and

Imbalanced Contexts Using Few-Shot and Contrastive Learning." Bioinformatics, 40(4), btae118. https://doi.org/10.1093/bioinformatics/btae118.

[46] Glen, R. C., Bender, A., Arnby, C. H., Carlsson, L., Boyer, S., & Smith, J. "Circular Fingerprints: Flexible Molecular Descriptors with Applications from Physical Chemistry to ADME." IDrugs, 9(3):199, 2006.

[47] Ma, H., Bian, Y., Rong, Y., Huang, W., Xu, T., Xie, W., Ye, G., & Huang, J. "Multi-View Graph Neural Networks for Molecular Property Prediction." arXiv:2005.13607v3 [q-bio.QM]. 12 June 2020.

[48] https://www.python.org

[49] Maddula, N. V. S. S. (2018, September 4). What Are the Key Features of Python. https://www.youth4work.com/talent/python/forum/124429-what-are-the-key-features-of-python

[50] https://numpy.org/

[51] https://pandas.pydata.org/

[52] http://rdkit.org

[53] https://github.com/stanford-futuredata/tensorflow-noscope

[54] https://www.w3schools.com/python/matplotlib_intro.asp

[55] https://keras.io/

[56] https://colab.research.google.com

[57] Lall, V. (2018, April 1). Google Colab — The Beginner's Guide. https://medium.com/lean-in-women-in-tech-india/google-colab-the-beginners-guide-5ad3b417dfa

[58] GCFGlobal. (n.d.). All about Google Drive. https://edu.gcfglobal.org/en/googledriveanddocs/all-about-google-drive/1/

[59] Mansouri, K., Grulke, C. M., Judson, R. S., & Williams, A. J. "OPERA Models for Predicting Physicochemical Properties and Environmental Fate Endpoints." Journal of Cheminformatics, 2018.