


الجمهورية الجزائرية الديمقراطية الشعبية

PEOPLE'S DEMOCRATIC REPUBLIC OF ALGERIA

وزارة التعليم العالي والبحث العلمي

MINISTRY OF HIGHER EDUCATION AND SCIENTIFIC RESEARCH

University Yahia farés Médéa — جامعة يحي فارس المدية

Faculty of sciences — كلية العلوم

قسم الرياضيات والاعلام الالي

DEPARTMENT OF MATHEMATICS AND COMPUTER SCIENCE

Specialty: Systems Engineering and Web Technology

Thesis for obtaining the Master's degree

Theme

Molecular Activity prediction using deep learning Graph Neural Network

Prepared and Presented by:

Kaci Asma

khélladi Chahrazed

Proposed and Directed by:

Dr. Hentabli Hamza

Academic Year: 2021-2022

Acknowledgment

We start first by thanking Allah for giving us power and patience and all of his blessings that made us achieve this degree.
We would also like to offer our sincere thanks to our advisor Dr. Hamza hentabli, at Yahia fares University, whose collective support, guidance and expertise have made this thesis a reality. We greatly thank him for sharing
knowledge and passion with us along this journey.

We also extend our gratitude to our families for their endless support, patience and inspiration during the long journey. They have always been and will continue to be our shelter and comfort during our academic period. We especially thank our parents for their love, support and prayers, those they have surrounded us with.

We also express our sincere thanks to our professors who provided us their knowledge and experience during our university years.

Dedication

I thank ALLAH who has given me all the will and patience to achieve this work.

To my dear parents, SLAMA NABILA and KHELLADI MEROUANE, who supported me emotionally with their blessings and prayers.

To my sisters LAMIA and SAMIRA, MY FAMILLY,

And to all

My friends and everyone I love with whom I share the most Beautiful moments.

To everyone who made me happy, even with a word or a word Smile, gave me strength to Continue

To supervisors, teachers and professors.

To my partner Asma and all her supportive family.

For anyone who reads and appreciates this work.

Chahrazed

Dedication

"Praise God, the one and only".

That this modest work, a little towards others bigger,

But he will always remain modest before your mercy.

I dedicated this modest work

To my dear mother and father, who always supported during my life, my studies, and encourage moving forward.

To My only sister "Hafsa" and

My brothers "Mohammed islem, Abdelmalek, and my little heart Racim".

To my partner (chahra),

For all family members without exception,

And To all my dear friends,

To all the promotion of 2nd year master's degree in Computer Science (2021/2022). To everyone who knows me.

Asma

Abstract

Basic structural features and physicochemical properties of chemical molecules determine their behavior during chemical, physical, biological and environmental processes and hence need to be investigated for determining and modelling the actions of the molecule. Computational approaches such as machine learning methods are alternatives to predict physiochemical properties of molecules based on their structures.

However, limited accuracy and error rates of these predictions restrict their use. In order to enhance the effectiveness of the similarity methods and QSAR measurement, the aim of this research is to develop a novel method of biological activity prediction, based on deep learning Graph neural network (GNN).

Key words:

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

Résumé

Les caractéristiques structurelles de base et les propriétés physicochimiques des molécules chimiques déterminent leur comportement au cours des processus chimiques, physiques, biologiques et environnementaux et doivent donc être étudiées pour déterminer et modéliser les actions de la molécule. Les approches informatiques telles que les méthodes d'apprentissage automatique sont des alternatives pour prédire les propriétés physicochimiques des molécules en fonction de leurs structures.

Cependant, la précision limitée et les taux d'erreur de ces prédictions limitent leur utilisation. Afin d'améliorer l'efficacité des méthodes de similarité et de mesure QSAR, l'objectif de cette recherche est de développer une nouvelle méthode de prédiction de l'activité biologique, basée sur l'apprentissage profond du réseau de neurones Graph (GNN).

Mots Clés:

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

ملخص

تحدد السمات الهيكلية الأساسية والخصائص الفيزيائية والكيميائية للجزيئات الكيميائية سلوكها أثناء العمليات الكيميائية والفيزيائية والبيولوجية والبيئية، وبالتالي تحتاج إلى التحقيق لتحديد ونمذجة إجراءات الجزيء. النهج الحسابية مثل طرق التعلم الآلي هي بدائل للتنبؤ بالخصائص الفيزيائية والكيميائية للجزيئات بناءً على هياكلها. ومع ذلك، فإن الدقة المحدودة ومعدلات الخطأ لهذه التنبؤات تقيد استخدامها. من أجل تعزيز فعالية طرق التشابه ، فإن الهدف من هذا البحث هو تطوير طريقة جديدة للتنبؤ بالنشاط البيولوجي، بناءً على شبكة **QSAR** وقياس الرسم البياني للتعلم العميق **(GNN)**

الكلمات الدالة

العلاقة الكمية بين البنية والنشاط - QSAR - التمثيل الجزيئي-التنبؤ بالنشاط البيولوجي – الشبكة العصبية للرسم البياني - GNN - التعلم العميق .

CHAPTER TITLE PAGE

ACKNOWLEDGMENTS i

DEDICATION ii

ABSTRACT iii

TABLE OF CONTENTS iv

LIST OF TABLES v

LIST OF FIGURES vi

TABLE OF SYMBOLS vii

LIST OF ABBREVIATIONS xi

LIST OF TERMINOLOGIES xii

1 Chapter 1: introduction

1. Introduction…….....1

2. Overview…….....1

3. Background of the problem……..…..... 2

4. Problem Statement…….....3

5. Objectives of the Research…….....5

6. Scope of the Research…….....5

7. Theses Out line……..…….....6

8. Summary…….……7

2 Chapter 2: LITERATURE REVIEW

1. Introduction…….…10

2. Definition of Chemoinformatics…….....11

3. Molecular Representations……11

3.1 1D descriptors……11

3.2 2D descriptors……12

3.3 3D descriptors……17

3.4 Other 3D descriptors……..18
4. Molecular similarity measurement……...19
4.1 the structure representation……20
4.2 the weighting scheme……..……...20
4.3 similarity coefficient……..……21
5. Quantitative structure- activity relationships……...…22
5.1 Quantitative structure- activity relationships definition……22
5.2 Brief history of (QSAR)…….23
5.3 Objectives development of QSAR…….23
5.4 The Steps of QSAR study……..24
5.5 QSAR modeling…….24
5.6 QSAR Development process……..25
5.7 Our QSAR Analysis can help you……..26
6. QSAR challenge……...26
6.1 Advantages and disadvantages of QSAR……..……27
7. Deep learning…….……..27
7.1 Machine learning…….……..27
7.2 Definition of deep learning…….....28
7.3 Convolutional neural networks……..……...32
7.3.1 Convolutional neural networks definition……31
7.3.2 Convolutional neural networks architecture…….……31
7.3.3 Advantages and disadvantages of CNN…….……..32
7.4 Graph neural networks.....34
7.4.1 Graph neural networks definition.....33
7.4.2 Applications of GNN……...34
7.4.3 Types of graph neural networks (GNN)…….……..…35
7.5 Graph convolution neural networks.....37
7.5.1 Definition of convolutional neural networks graphs……37
7.5.2 Architecture of convolutional neural networks ……...……37
7.5.3 Application of convolutional neural networks ……...….38
8 Discussion.....39
9 Summary……..……40

3 Chapter 3: RESEARCH METHODOLOG

1. Introduction…….….42

2. Research Framework……42

2.1 Research plan and review of literature…….....……44

2.2 Collection of chemical dataset……...…44

2.3 Converting molecule data from molecule descriptors……...….45

3. Ensemble learning……..…48

3.1 application of information retrieval similarity using tanimonto…...…...48

4. Design and steps…….….....50

4.1 Global Architecture…….....50

4.2 Find graph structure…….....…52

4.3 Specify graph type and scale……...……52

4.4 Design loss function…….…...…53

4.5 Build model using computational modules…….....53

5 Performance Evaluation……...…54

6 Benchmarking……..……55

7 Summary……..…...56

4 Chapter 4: implementation result and discussion

1. Introduction…….….59

2. Development environment……...59

2.1 python…….59

2.2 python features……...59

2.3 python module used……...…61

2.4 Google colaboratory…….…...…63

2.4.1 Google colaboratory definition…….…….…..…..63

2.4.2 benefits of colab…….64

2.5 Google drive……..65

3. Explanation of our approach……66

3.1 Dataset……..……..66

3.2 Data download……..…….....66

4. Data preprocessing…..…….67

4.1 splitting the data……...……..67

4.2 train ,validation and test…….67
4.3 network architecture…….……..68
4.3.1 GNN implementation…….…..…..68
4.4 Evaluation……...71
4.1.1 Train the GNN model…….....71
5. Summary……..…72

List of figures:

Figure 1: Example of generation of a fingerprint.

Figure 2: Example of fragment types used in 2D fingerprints.

Figure 3: Representation of molecular descriptors.

Figure 4: general steps involved in QSAR modeling.

Figure 5: Types of Machine leaning.

Figure 6: Difference between Machine Learning and Deep Learning.

Figure 7: Example of A CNN sequence to classify handwritten digits.

Figure 8: A graph neural network architecture design.

Figure 9: Difference between graph and node classification.

Figure 10: Architecture of Recurrent Graph Neural Network.

Figure 11: Image a Comprehensive Survey on Graph Neural Networks Left: Convolution on a regular graph such as an image. Right: Convolution on the arbitrary graph structure.

Figure 12: Architecture of the convolutional neural network.

Figure 13: Flow Chart of the Framework.

Figure 14: Schematic illustration of the construction of initial graphs from the smile of data base.

Figure 15: Global architecture of QSAR model.

Figure 16: The general design pipeline for a GNN model.

Figure 17: Google drive applications and functions.

Figure 18: Dataset example from the dataset.

Figure 19: Split dataset to train /test.

Figure 20: Implement train and evaluate experiment.

Figure 21: compile and trains an input model.

Figure 22: The model during of training.

Figure 23: Train the GNN model.

Figure 24: Epochs the GNN model.

Figure 25: The training and validation loss.

Figure 26: Evaluate the baseline model on the test data split.

Figure 27: The result of evaluation.

Figure 28: Comparison between train and test for accuracy and for loss in GNN.

List of tables:

Table 1: 2D versus 3D structure representation.

Table 2: The most commonly used similarity or distance coefficients for similarity searching in chemical database.

Table 3: The Tanimoto Similarity Measures.

Table 4: Diversity between machine learning (ML) and deep learning (DL).

Table 5: Activity classes and sizes of active datasets derived from the MDDR

Table 6: atom feature used to define the initial node in the molecular graphs.

Table 7: Bond features used to define the initial edges in the molecular graphs.

Table 8: The Tanimoto Similarity Measures.

Table 9: Our data and It's activities.

List of Acronyms

QSAR Quantitative structure-activity relationship

ML machine learning

DL deep learning

1D One Dimensional

2D Two Dimensional

3D Three Dimensional

GNN graph neural network

CNN convolutional neural network

GCNN graph convolutional neural network

MDL Molecular Design Limited

MDDR MDL Drug Data base Report

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.

Similarity Searching

- Standard tool for drug discovery .A chemical compound retrieval system that requires

specification of an entire target structure, which is compared with the corresponding sets of features for each of the database structures. The database molecules are then sorted in decreasing order of similarity with the target.

General Introduction

General Introduction

Chemoinformatics is a new field that appeared a few years ago. It concerns the development, creation, organization, storage, dissemination, analysis, visualization and use of chemical information, it represents a field in rapid evolution. Currently, the specific methods of chemoinformatics have become essential for the development of new compounds, materials and processes.

Chemoinformatics is a scientific discipline that has evolved a lot over the past 40 years. This science is at the interface between chemistry and computer science. It has been realized that in many areas of modern chemistry, the vast amount of data and information provided by chemical research can only be analyzed by computational methods. Moreover, the new problems encountered in chemistry are so complex that the use of computers has become essential. This is why methods have been developed to predict the chemical, physical and biological properties of materials and chemical compounds. These methods have also been developed for the representation of drugs, the elucidation of crystal structures, in the prediction of chemical reactions and in the representation of organic molecules.

Quantitative structure—activity relationships (QSARs) attempt to correlate chemical structure with activity using statistical approaches. The QSAR models are useful for various purposes including the prediction of activities of untested chemicals. Quantitative structure—activity relationships and other related approaches have attracted broad scientific interest, particularly in the pharmaceutical industry for drug discovery and in toxicology and environmental science for risk assessment. An assortment of new QSAR methods have been developed during the past decade, most of them focused on drug discovery. Besides advancing our fundamental knowledge of QSARs, these scientific efforts have stimulated their application in a wider range of disciplines, such as toxicology, where QSARs have not yet gained full appreciation. Chemical similarity (or molecular similarity) refers to the similarity of chemical elements, molecules or chemical compounds with respect to either structural or functional qualities, i.e. the effect that the chemical compound has on reaction partners in inorganic or biological settings. Biological effects and thus also similarity of effects are usually quantified using the biological activity of a compound. In general terms, function can be related to the chemical activity of compound.

The deep learning being a subcategory of the machine learning follows the human instincts of learning by example to produce accurate results. The deep learning performs training to the computer frame work to directly classify the tasks from the documents available either in the form of the text, image, or the sound. Most often the deep learning utilizes the neural network to perform the accurate classification and is referred as the deep neural networks; one of the most common deep neural networks used in a broader range of applications is the graph neural network that provides an automated way of feature extraction by learning the features directly from the graphs unlike the machine learning that extracts the features manually. This enables the deep learning neural networks to have a state of art accuracy that mostly expels even the human performance.

Graph neural networks are trainable functions which operate on graphs—sets of elements and their pairwise relations—and are a central method within the broader field of geometric deep learning. They are very expressive and have demonstrated superior performance to other classical deep learning approaches in a variety of domains. The data in particle physics are often represented by sets and graphs and as such, graph neural networks offer key advantage.

Chapter 1: Introduction

1. Introduction……1

2. Overview…….....1

3. Background of the problem……..…..... 2

4. Problem Statement…….....3

5. Objectives of the Research…….....5

6. Scope of the Research…….....5

7. Theses Out line……..**.....**6

8. Summary……7

Chapter 1

1. Introduction

With the development of high-speed information technology in various fields, especially informatics, and the human need to adapt to this speed, it was necessary to develop and innovate.

Quantitative structure–activity relationship (QSAR) models are routinely applied in the drug discovery process, which are classification models that predict the biological activities of molecules based on the features derived from their molecular structures. These models are usually used to prioritize a list of candidate molecules for future laboratory experiments and to help chemists gain better insights into how structural changes affect a molecule's biological activities.

Many underlying relationships among data in several areas of science and engineering, e.g., computer vision, molecular chemistry, molecular biology, pattern recognition, and data mining, can be represented in terms of graphs. In this paper, we propose a new neural network model, called graph neural network (GNN) model that extends existing neural network methods for processing the data represented in graph domains. This GNN model, which can directly process most of the practically useful types of graphs.

In this chapter, develop a novel method of biological activity prediction based QSAR approach based on deep learning Graph neural network (GNN), is presented. 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 similarity searching of molecular structure is closely related to the Similar Property Principle, whereby, molecules that are structurally similar are likely to have similar properties. This relationship lead to the range of chemoinformatics techniques as it had established a discipline that has applications in many areas of chemistry and computational molecular science such as molecular diversity analysis, clustering, QSAR methods and also similarity searching . In order to conduct the similarity searching.

3. Background

Recent advances in computer hardware and software have led to a revolution in deep neural networks that has impacted fields ranging from language translation to computer vision. Deep learning has also impacted a number of areas in drug discovery, including the analysis of cellular images and the design of novel routes for the synthesis of organic molecules. While work in these areas has been impactful, a complete review of the applications of deep learning in drug discovery would be beyond the scope of a single Account. In this Account, we will focus a one key areas where deep learning has impacted molecular design: the biological activity prediction of molecules.

Quantitative structure activity relationship is study of relationship between physiochemical (independent) properties and biological (dependent) activity of bioactive molecules. It plays an important role in the drug discovery and development and involves steps including data preparation, data analysis, and model validation. Validation is important process in QSAR analysis. At present, QSAR methodologies comprise a series of possibilities, including joining forces with machine learning methods and increasing even more the swiftness they might bring to the prospective development of therapeutics in the Health Sciences scope.

One of the most significant advances in the development of quantitative structure–activity relationships (QSARs) has come from the application of deep learning methods to the prediction of the biological activity and physical properties of molecules in drug discovery programs. Rather than employing the expert-derived chemical features typically used to build predictive models, researchers are now using deep learning to develop novel molecular representations. These representations, coupled with the ability of deep neural networks to uncover complex, nonlinear relationships, have led to state-of-the-art performance. While deep learning has changed the way that many researchers approach QSARs, it is not a panacea. As with any other machine learning task, the design of predictive models is dependent on the quality, quantity, and relevance of available data.

We aim to efficiently learn distributed representation of graph, that is, a map that turns variable-size graphs into fixed size vectors or matrices. Such a representation would benefit greatly from a powerful pool of data manipulation tools. This rules out traditional approaches such as graph kernels and graph feature engineering, which could be either computation or labor intensive.

The biggest difficulty for machine learning with molecules is the choice and computation of “descriptors”. Graph neural networks (GNNs) are a category of deep neural networks whose inputs are graphs and provide a way around the choice of descriptors. A GNN can take a molecule directly as input.

Graph neural networks are proposed to combine the feature information and the graph structure to learn better representations on graphs via feature propagation and aggregation. Due to its convincing performance and high interpretability, GNN has recently become a widely applied graph analysis too. GNNs can do what Convolutional Neural Networks (CNNs) failed to do.

Another challenge in biological activity prediction is that the screening process is time-consuming, limiting the number of molecules to be tested. Traditional training of a small dataset with deep neural networks might cause over fitting. To improve the prediction performance, work has been proposed to use graph neural networks to graphs can also be good sources of data to train machine learning models for complicated tasks, which are a type of machine learning algorithm that can extract important information from graphs and make useful predictions, GNNs have been shown to be an effective tool for solving issues that can be represented as graphs.

4. Problem statement

The basic idea behind the QSAR approach is the property of similarity, which is the principle that structurally similar molecules are more likely to exhibit activities similar to structurally similar molecules. Currently, the concept of molecular similarity has been widely used in rational drug design. This, which is the search for similar functional molecules, is one of the most important goals in drug design, and this can be achieved by searching databases for structurally similar molecules. However, molecular similarity is not always as we think but rather an external

criterion that determines similarity depends on it. Several similarity methods have been proposed in the literature, which can be categorized as 2D similarity methods and 3D similarity methods. However, while 2D similarity methods are well suited for identifying nearby isotopes, the biological activity that has led to the development of various similarity search methods consider 3D representation of molecules (3D similarity methods). These approaches are of particular importance, if there is a need to define new leadership structures. However, 3D similarity methods have been rejected as alternatives to random sorting, and 2D structure-based tools are becoming simpler and more suitable for a high-throughput model for drug discovery. The superiority of 2D methods is due to their simplicity, ease of use, and evidence of the ability of 2D descriptors to explain structure activity data, Thereafter it is recommended to use several similar methods where possible. Several studies in the field of information retrieval have shown that retrieval models exist. Inference networks have significant improvements in retrieval performance compared to traditional models. In the field of chemical informatics, there are several techniques that arise from the field of information retrieval, as many similarities have been identified between them. These comparisons provided and using the tanimoto coefficient the effective method the molecular similarity searching in basis for our work.

Neural networks were widely used for quantitative structure–activity relationships (QSAR) in the 1990s. Because of various practical issues (e.g., slow on large problems, difficult to train, prone to over fitting, etc.) and for this problem We propose a new type of individual QSAR classifier that is deep neural network model based on graphs neural networks (GNNs), It automatically extracts sequential features from a simplified molecular-input line-entry system (fingerprints),along with her ability of to reveal complex and nonlinear relationships, have led to sophisticated performance model effective for similarity methods and QSAR measurement.

GNNs is from the best models used deep learning for biological activity prediction molecular , a model which builds a set of learning models and combines multiple models to produce final and effective predictions. Theoretically and empirically, it has been shown that the predictive power of ensemble learning surpasses that of a single individual learner if the individual algorithms are accurate and diverse.

4. Research objective

The main goal of this research is to develop a novel similarity-based QSAR approach using graph neural networks, with ability to develop a novel method of biological activity prediction, based on deep learning Graph neural network (GNN).

To achieve this goal, the following objectives have been set:

a) To retrieve the activity of compounds from chemical databases using RDKit Library.
b) Development of predictive and robust QSAR for unkown molecule using GNN.
c) To prove that GNNs is an effective tool for solving issues that can be represented as graphs.

5. Research scope

Quantitative structure activity relationship is study of relationship between physiochemical (independent) properties and biological (dependent) activity of bioactive molecules. It plays an important role in the drug discovery and development and involves steps including data preparation, data analysis, and model validation. Validation is important process in QSAR analysis.

Recently, a machine learning method, graph neural networks (GNNs), has shown promising results for the prediction of structure–property relationships. GNNs utilize a graph representation of molecules, where atoms correspond to nodes and bonds to edges containing information about the molecular structure. More specifically, GNNs learn physicochemical properties as a function of the molecular graph in a supervised learning setup using a back propagation algorithm. This end-to-end learning approach eliminates the need for selection of molecular descriptors or structural groups, as it learns optimal fingerprints through graph convolutions and maps the fingerprints to

the physicochemical properties by deep learning. We develop GNN models for predicting biological activity.

This study will focus on similarity methods based on 2D fingerprints. These methods are used to graphically determine the degree of similarity between a pair of molecules characterized by two-dimensional fingerprints. These methods are applied with non-2D fingerprints and the Tanimoto coefficient. Similarity searching of molecular structure has been an important application in the Chemoinformatics, especially in drug discovery. Similarity searching is a common method used for identification of molecular structure. It involve three main principal component of similarity searching: structure representation; weighting scheme; and similarity coefficient. Inthis paper, we introduces Weighted Tanimoto Coefficient based on weighted Euclidean distance in order to investigate the effect of weight function on the result for similarity searching. The Tanimoto coefficient is one of the popular similarity coefficients used to measure the similarity between pairs of the molecule. The most of research area found that the similarity searching is based on binary or fingerprint data.

The study developed a new method for molecular similarity in the graph of neural networks. This method evaluates two-dimensional fingerprints or others and a large dataset derived from the Drug Data Report (MDDR), when a single reference structureor multiple reference structures. The performance of this methods is evaluated against the performance Traditional similarity methods.

6. Theses out line:

This thesis consists of four main parts, with the exception of the introductory chapter. While the first two parts describe the background in addition to what was previously published Work in the field of molecular measurement, part three describing the research Methodology of work in this thesis.

Finally, the last part presents results and description of the molecular similarity method by means of a graph neural network. **Chapter Two,** Molecular Similarity Review, begins with an overview of chemical information and the different types of molecular similarity measurement, and we discuss the implementation of molecular

similarity deep learning techniques. In the third section, we present molecular representations, which can be used for molecular similarity research as well as molecular analysis. Here, we also describe in detail fingerprint-based similarity methods and different types of transactional similarity. In addition, we talk about the quantitative relationship between structure and activity. This chapter concludes with a discussion and summary of the applicability of the described methods for searching for molecular similarities.

Chapter 3, Research Methodology, describes the overall methodology adopted in this research to achieve the objectives of this work. Here, we try to give a general picture about each phase in our research framework.

Chapter 4 introduces python and his libraries, Google_Colab, and discussions in this project.

Chapter 5, 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.

7. Summary:

In this chapter, we try to give a broad overview of the problems involved in biological activity prediction which is the subject of our work. This chapter serves as an introductory to the research problem addressed in this work. The goal, objectives, the scope, and the outline of this work were presented.

Chapter 2:

LITERATURE REVIEW

Table of content:

1. Introduction……10

2. Definition of Chemoinformatics…….....11

3. Molecular Representations……11

3.1 1D descriptors……11

3.2 2D descriptors……12

3.3 3D descriptors……..……..……17

3.4 Other 3D descriptors……..…....19

4. Molecular similarity measurement……...19

4.6 the structure representation…….……20

4.7 the weighting scheme……..……...20

4.8 similarity coefficient…….……..…..……21

5. Quantitative structure- activity relationships……..……...…23

5.1 Quantitative structure- activity relationships definition…….……23

5.2 Brief history of (QSAR)…….……..23

5.3 Objectives development of QSAR…….……24

5.4 The Steps of QSAR study……..……...24

5.5 QSAR modeling……...……25

5.6 QSAR Development process……...……..26

5.7 Our QSAR Analysis can help you……...……..27

6. QSAR challenge……...27

6.1 Advantages and disadvantages of QSAR……..……28

7. Deep learning…….…..28

7.1 Machine learning……..…….…..28
9.2 Definition of deep learning…….....30
9.3 Convolutional neural networks……...……...32
9.3.1 Convolutional neural networks definition……32
9.3.2 Convolutional neural networks architecture……32
9.3.3 Advantages and disadvantages of CNN……..33
9.4 Graph neural networks.....34
9.4.1 Graph neural networks definition.....34
9.4.2 Applications of GNN……...35
9.4.3 Types of graph neural networks (GNN)…….……..…36
9.5 Graph convolution neural networks.....38
9.5.1 Definition of convolutional neural networks graphs……38
9.5.2 Architecture of convolutional neural networks ……...……38
9.5.3 Application of convolutional neural networks ……...….39
9. Discussion.....40
10. Summary…….41

1. Introduction

Quantitative Structure-Activity Relationship (QSAR) models are becoming one of the most interesting fields for developing the rapeutics and therapeutics related patents. At present, QSAR methodologies comprise a series of possibilities, including joining forces with machine learning methods and increasing even more the swiftness they might bring to the prospective development of therapeutics in the Health Sciences scope.

Quantitative structure–activity relationship (QSAR) models are routinely applied computational tools in the drug discovery process. Thus, QSAR models are regression or classification models that predict the biological activities of molecules based on the features derived from their molecular structures. These models are usually used to prioritize a list of candidate molecules for future laboratory experiments and to help chemists gain better insights into how structural changes affect a molecule's biological activities. Developing accurate and interpretable QSAR models is therefore of the utmost importance in the drug discovery process. Deep neural networks, which

are powerful supervised learning algorithms, have shown great promise for addressing regression and classification problems in various research fields, including the pharmaceutical industry.

In this chapter, we briefly review the applications of deep neural networks in QSAR modeling, the purpose is to give a comprehensive and detailed idea of several concepts:

Domain of Chemical Informatics, Deep Learning, Diagram of Neural Networks. For deep learning, some information about the concept of deep learning is provided in the form of simplified definitions and types, how they work, as well as application areas. Second, the neural network graph (GNN) has its definition, history, process, and various approaches. Third, the quantitative structure-activity relationship (QSAR) is dealt with its definition, history and how it works.

2. Chemoinformatics:

2.1 Definition of Chemoinformatics:

Chemoinformatics, sometimes known as chemoinformatics or chemical informatics is the field of the use of computer technologies for processing of chemical data and information. The term “chemoinformatics” was first used as recently as 1998 but the activities incorporated in the term had been practiced in drug industries long before that.

Today the term “chemoinformatics” seems to have become the favored term used for example by the Journal of chemoinformatics. Mostly chemoinformatics is about chemical structures in one way or another. Among the future challenges where chemoinformatics might play an important role are:

- To provide efficient in silico tools for predicting Absorption, Distribution, Metabolism, Excretion, and Toxicity (ADMET) properties.
- To help overcome the stalled drug discovery by efficient in silico experiments in order to prioritise for in vitro experiments.

3. Molecular Representations:

The Molecular descriptors can be divided into 1D, 2D descriptors (2D fingerprint) and 3D descriptors.

3.1 1D Descriptors:

One D descriptors are a single numeric values assigned to the molecule based on certain aspects of the molecule. These values are usually derived from computed physicochemical properties such as Polaris ability, volume, molecular weight, hydrophobicity, and refractive index. They are usually used to predict the physical properties of the molecules since no geometrical information is contained. These descriptors have been used in applications such as clustering of compounds databases and database comparisons. Several similarity measures make use of the global physicochemical properties of whole molecules since the relevant property, in most cases, is unknown and can only be a combination of different one-dimensional properties. Examples of these types of properties as investigated by CAS workers are ClogP (Log of the octanol / water partition coefficient) and molar refractivity (the ratio of the speed of light in a vacuum to its speed in a sample compound).

Some of these properties are very important in determining the activity and transport of drugs such as hydrophobicity. A molecule's hydrophobicity can affect how tightly it binds to protein and its ability to pass through a cell membrane. Other whole-molecule properties include principal moments of inertial, principal axes, volume of the inertial ellipsoid, molecular weight, etcetera. Calculating some of these properties is considered computationally expensive such as properties that are calculated using quantum mechanics packages.

Hydrophobicity governs numerous and different biological processes, such as, for example, transport, distribution, and metabolism of biological molecules; molecular recognition; and protein folding. It is modelled using the logarithm of the partition coefficient between n-octanol and water (logP) . In order to enhance the molecular characterization, 1D descriptors are sometimes combined with 2D descriptors. For instance, Kearsley and his colleagues have generalised the atom pair and topological torsion descriptors by using physicochemical properties in place of atom types.

3.2 2D Descriptors:

A large variety of 2D structural descriptors is used to represent the structure of organic compounds in a numerical form. Among these, descriptors computed based on molecule graph are widely used in modelling physical, chemical, or biological properties. The simplest 2D descriptors are based on simple counts of features such as hydrogen donors, hydrogen bond acceptors, ring systems (such as aromatic rings) and rotatable bonds, whereas the complex 2D descriptors are computed from a complex mathematical equations such as 2D fingerprints and topological indices.

3.2.1 2D Fingerprints:

Among the large number of descriptors, 2D fingerprints are the most commonly used, where they were initially developed to provide a fast screening step in substructure searching. Furthermore, 2D fingerprints have been found to enhance the effectiveness of retrieval when used in similarity searching. Fingerprints are generated by fingerprinting process. Fingerprinting is the process of converting a chemical structure (in the form of a connection table) into a string of “0”s and “1”s (binary form). This binary form represents a kind of chemical shorthand which identifies the presence or absence of some structural feature in the molecule as shown in Figure 2.2. There are two different types of 2D fingerprints: dictionary based fingerprints and hashed fingerprints. The length of the fingerprints in dictionary-based is restricted to the number of fragments in the dictionary, where each bit position in binary string (fingerprint) often is assigned to only one specific substructure fragment in a dictionary. In some situation, some restricted bits can make the interpretation of the results more straightforward, for instance, if specific bits are important to explain the observed activity.

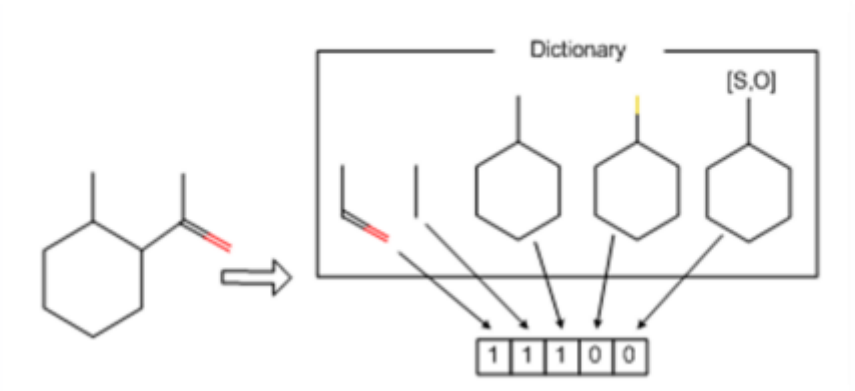


Figure 1: Example of generation of a fingerprint.

In dictionary-based fingerprints, a molecule is break up into fragments of specific functional groups or substructures. However, only those fragments (structural keys) recorded in a predefined fragment dictionary are used to specify the corresponding bit positions of the fragments in the fingerprint. Bits represent the presence or absence of fragments either individually or as a group. CAS ONLINE Screen Dictionary for substructure searching, Barnard Chemical Information system, and MDL MACCS key system are examples of dictionary-based fingerprints.

The first step in dictionary-based fingerprinting is assembling the fragment dictionary which is very subjective to the type of the database. However, this step can be achieved through analysis of a database to generate different types of fragments. Analysing of a database typically produced tens or hundreds of thousands of distinct fragments, where many fragments are very frequent and few fragments are very infrequent. The fragments that occur infrequently may be more likely to be useful than fragments which occur very frequently. These fragments can involve atoms, bonds and rings. Examples of these types of fragments are augmented atoms, atom sequences, ring sequence, ring fusion sequence, atom pair and topological torsion,These fragments are illustrated in Figure 2.

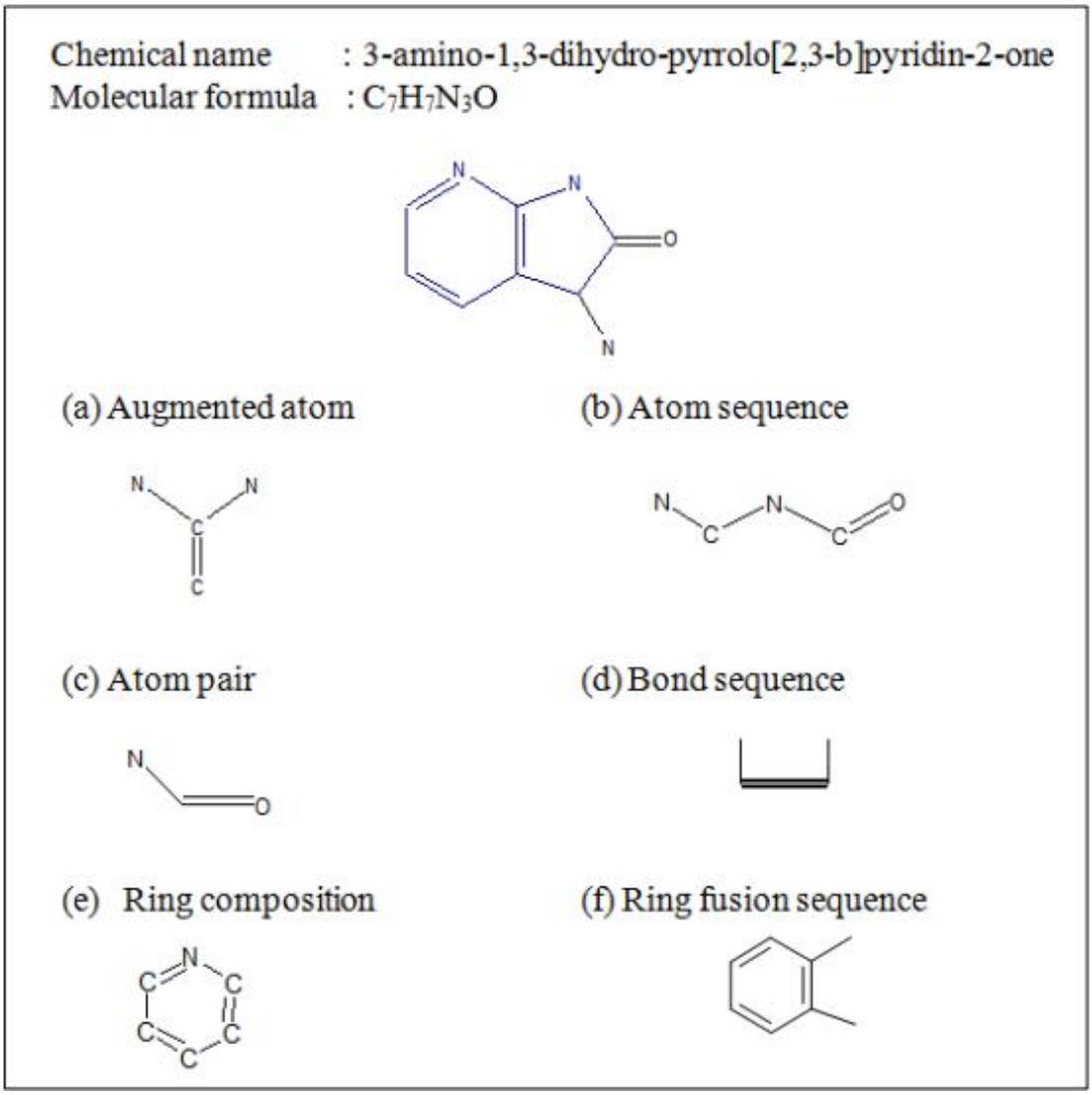


Figure 2: Example of fragment types used in 2D fingerprints.

The co-occurrence of the fragment might also introduce unwanted bias into fragment-based similarity measures by placing unwanted emphasis on certain features. For example, setting a bit for a 5-carbon path will also set bits for 1, 2, 3 and 4 carbon paths .Several methods have been introduced to reduce the number of the fragments to fit the fixed-length fingerprint and also retain the fragments that act as the best descriptors. These methods including statistical analysis of the fragments frequencies to remove the very frequent/infrequent fragments, generalisation of specific fragments to less specific forms that cover many related fragments and assignment of the same screen number to several different but related fragments Recently, Xue and co-worker have introduced short fingerprints called "minifingerprints" that are constructed by omitting structural fragments that were

over-represented or under-represented in two large compound collections. They found that this minifingerprint performed better than the original 2D fingerprint.

The alternative to dictionary-based fingerprint is to use a hashed fingerprint. One of the disadvantages of dictionary-based fingerprints is the optimum set of fragment that is often data set dependent. On the other hand, the hashed fingerprint is not dependent on a pre-defined fragment dictionary and so in principle it has the advantage of being applicable to any type of molecular structure (any fragment ispresent in the molecule will be encoded). The hashed fingerprint of a molecule is bit string (binary form) that contains information on the structure. In hashed fingerprints, all the unique fragments that exist in a molecule are hashed using some hashing function to fit into the length of the bit string. This approach allows for more generalisations because it does not depend on a predefined list of structural fragments, and the generated fingerprints are characterised by the nature of the chemical structures in the database rather than by the fragments in some predefined list. Instead of using a fragment dictionary, a hashed fingerprint is produced by generating all possible linear paths (linear patterns) of connected atoms through the molecule containing a defined number of atoms (typically seven). For example, the molecule OC=CN (SMILES representation) would generate the C, O, N for 0-bond paths, OC, C=C, CN for 1-bond paths, OC=C, C=CN for 2-bond paths, and OC=CN for 3-bond paths. When using a hashing function, a given number of bits (typically four or five) in the fingerprint bit string is set for each pattern. It is possible that the same bit is set by multiple patterns that because there is no pre-defined set of pattern and because the number of possible patterns is so huge. This phenomenon is called bit collision. Few bit collisions in the fingerprint is tolerable, but too many may result in losing information in the fingerprint.

Hashed fingerprints are particularly associated with the Daylight Chemical Information Systems Inc. and Tripos systems. In Daylight systems, all paths up to 7 bonds long are detected (path-based approach) and set the corresponding bits to “1” in the fingerprint for each present path. The resulting bit is then folded to reduce storage requirements and to speed searching. The UNITY system from Tripos Inc. uses a combination of the structural keys (dictionary-based fingerprints) and hashed fingerprints.The hashed fingerprints have several advantages, since fingerprints have no pre-defined set of patterns, one fingerprinting system serves all databases and all

types of queries (generality) and that often leads to superior database screening and more efficient 2D and similarity searching. However, bit collision (overlapping of patterns) can cause loss of information and give false similarity values. Any individual bit does not actually represent anything unique about the structure of the molecule. Adding all the fragments could also give problems by including many co_occurring fragments and by including large numbers of frequently occurring fragments in the dataset, which could reduce the effects of other fragments that are more meaningful to the similarity measure. Folding the bit strings to make it even smaller can make the problem worse. Folding can also result in very small and dense fingerprints.

The 2D fingerprints described above effectively encode the substructures present in a molecule. It would not seem unreasonable that the proportion of substructures in common between two molecules should be a reasonable measure of similarity of the overall molecules. In mathematical terms this is a comparison of the bits in the fingerprints which are set on.

Originally, 2D fingerprint was designed to improve the performance of the substructure searching algorithms and overcome the shortcomings of the structural keys rather than to be used as molecular descriptors. Recently, similarity measures based on bit strings (2D fingerprint) are one of the most widely used approach for database searching. Furthermore, 2D fingerprint searching is used in the selection of compounds for inclusion in biological screening programs. The 2D fingerprints are often powerful similarity search tools, and even simple search strings and atom count vectors recognize active compounds successfully and is also found to be effective in clustering application (grouping active molecules together). Therefore, it is not surprising that similarity searching based on 2D fingerprint continues to be popular in chemoinformatics research. The popularity of using the simple 2D fingerprint for similarity-based virtual screening is not only because it’s computational efficiency but also for its demonstrated effectiveness in the many comparative studies that have been carried out .

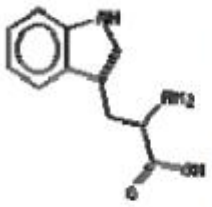
3.3 3D Descriptors:

3D descriptors are molecular descriptors derived from a geometrical representation. Typically, 3D descriptors provide more information and

discrimination power than topological descriptors for similar molecular structures and molecule conformations, and that because the geometrical representation involves knowledge of the relative positions of the atoms in 3D conformation.

Despite their higher information content, 3D descriptors also have disadvantages, such as, calculating 3D descriptors is more computationally expensive than the processing of 2D descriptors, most 3D descriptors (e.g. grid-based) require alignment rules to achieve molecule comparability, and the complexity (for deriving 3D descriptors) can increased significantly when several molecules conformations instead of a single molecule conformation is taken into account. For these reasons, 2D descriptors and other simple descriptors are preferred for screening large database of molecules. On the other hand, 3D descriptors can be efficiently searched for relationship between the molecular structures and complex properties such as biological activity, by exploiting their large information content .Examples of 3D descriptors are 3D screens, potential-pharmacophore-point descriptors, affinity fingerprints, 3D atom environment for use in atom mapping similarity searching and 3D molecular fields for use in field-based similarity searching. Shape descriptors, like surface area and volume, also use the 3D shape of the whole molecule.Table below shows some relative merits and problems of 2D and 3D structural representations.

Table 1: 2D versus 3D structure representation.

2D		3D	
	**Pros:** • Show complete structure • Easy recognition of patterns • Chemists know how good structures look like **Cons:** • Removes too much information from the real structure • Make impossible spatial matching of structures		**Pros:** • All available structural information are present • Understand shapes • See what would be hidden in a 2D view **Cons:** • Limited to viewing part of structure • Unsuited for quick comparisons • Needs interaction to avoid ambiguities

3.4 Other 3D Descriptors:

Other descriptors such as, 3D descriptors which are used in alignment-based similarity searching and 3D atom environment descriptors which are used in atom mapping methods . In the atom mapping method, each atom in one molecule is compared with each atom in a second molecule in manner similar to MCS procedure but in a 3D environment. The resulting list of interatomic similarities is used to identify pairs of geometrically similar atoms, which are used as descriptors in the

calculation of the overall intermolecular similarity. Typically, the intermolecular similarity is given in terms of the mean similarity per atom in the query molecule or the smaller molecule of the two compared molecules.

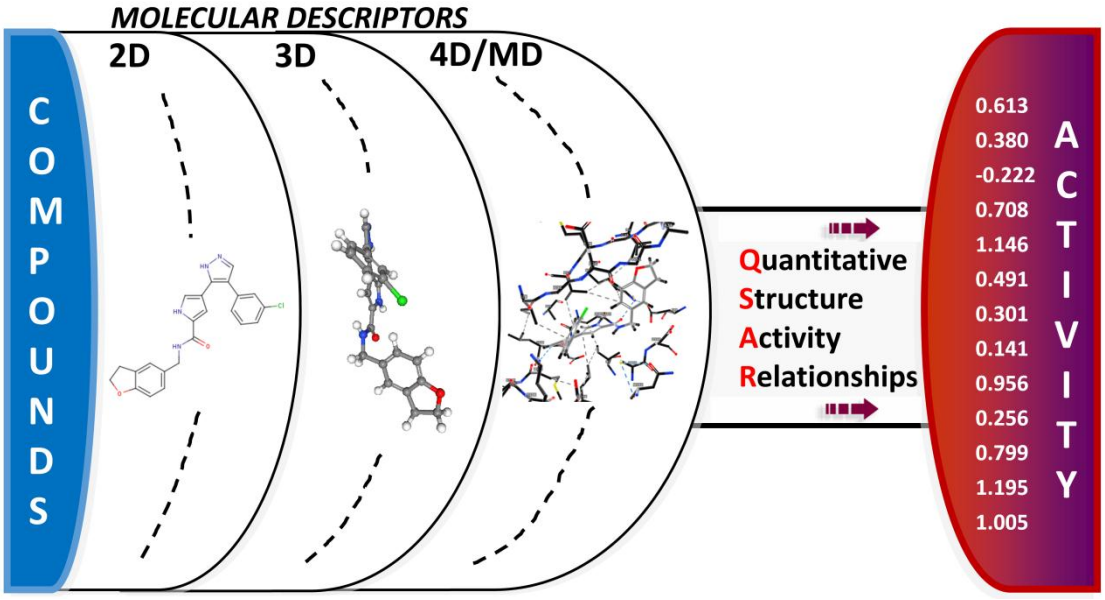


Figure 3: representation of molecular descriptors.

2. Molecular similarity measurement

The similarity searching of molecular structure is closely related to the Similar Property Principle, whereby, molecules that are structurally similar are likely to have similar properties. This relationship lead to the range of cheminformatics techniques as it had established a discipline that has applications in many areas of chemistry and computational molecular science such as molecular diversity analysis, clustering, QSAR methods and also similarity searching . In order to conduct the similarity searching.

There are three important components of similarity measures that need to be considered. First, the structure representation (used to characterize molecules), second, the weighting scheme (used to reflect the relative importance of different parts of the representation), and lastly is similarity coefficient. The similarity coefficient that is used to quantify the degree of resemblance between two

appropriately weighted structural representations. We will provides a detailed explanation of the three components.

2.1 The structure representation :

The structure representation used to characterize molecules, they are commonly divided into three classes: whole molecule (sometimes called 1D descriptors; descriptors that can be calculated from 2D representations of molecules; and descriptors that can be calculated from 3D representations. Whole molecule descriptors are measured or computed single numbers that describe bulk molecular properties, such as the number of stereo-centres, the molecular weight, the octanol-water partition coefficient or the polar surface area and for similarity purposes.

2.2 The weighting scheme:

A weighting scheme is used to differentiate between different features in a molecule, based on how important they are in determining the similarity of that molecule with another molecule. Certain molecular features can be emphasised by associating higher weights with them when calculating similarity. Different types of statistical information can be extracted from computerized representations of molecules to form the basis for a fragment weighting scheme. For example, greater weights can be given to: fragments that occur frequently in a molecule, fragments that occur in small molecules and also fragments that occur less frequently in a data set.

Different types of statistical information can be extracted from computerized representations of molecules to form the basis for a feature weighting scheme. These are as follow, (a) Feature Frequency (ff), the number of occurrence of a particular feature within a compound, with more frequently occurring features being given greater weights than those that occur less frequently. (b) Inverse Compound Frequency (icf), the frequency of the feature in the whole compound collection, with less frequently occurring feature being given a greater weight than those that occur more frequently throughout the molecule collection. (c) Compound size (compound length), the number of the unique features assigned to a compound, with features in a smaller compound being assigned a greater weight than the same features in a larger compound.

2.3 Similarity coefficient:

Many different types of similarity coefficients have been developed to obtain a numeric value to the degree of similarity between the two objects in the different fields we review only those coefficients have been used in chemical Four main types of similarity coefficients are available in the literature: association coefficients, correlation coefficients, distance coefficients, and probabilistic coefficients. Most of the coefficients have the ability to deal with the binary and continuous (real or integer value) descriptors where different versions of each coefficient are used. As shown that the Tanimoto coefficient, an example of the class of coefficients known as association coefficients, provided an effective way of comparing 2D fingerprints. Table 1. Shows the most commonly used in chemical information systems.

Table 2: 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

Let A and B two molecules, described by means of vectors WA and WB of N features, where SAB is the similarity between molecules A and B. For molecule described by bit-string (binary form) descriptors, calculate the degree of similarity depend on:

- a, b are the number of bits set to "1" in the bit-string of molecules A and B respectively.
- c is the number of bits set to "1" in the both bit-string of molecules A and B.

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 3: 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

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

5. Quantitative Structure-Activity Relationships

5.1 Quantitative structure-activity relationship definition:

Quantitative structure-activity relationship (QSAR) is a computational or mathematical modeling method to reveal relationships between biological activities and the structural properties of chemical compounds. The underlying principle is that variations in structural properties cause different biological activities.

QSAR modeling helps prioritize a large number of chemicals in terms of their desired biological activities and, as a result, significantly reduces the number of candidate chemicals to be tested with in vivo experiments.

QSAR have an essential role in drug design process these days, because they are cheaper alternative than the medium throughput in vitro and low throughput in vivo assays which also, in drug discovery and environmental toxicology, QSAR models are now regarded as a scientifically credible tool for predicting and classifying the biological activities of untested compounds, drug resistance, toxicity prediction and physicochemical properties prediction.

7.1 Brief history of (QSAR):

Quantitative structure-activity/property relationships (QSAR/QSPR) are increasingly used, due to the growth of means of calculation. They have been widely used in the pharmaceutical, chemical and cosmetic industries, especially for the rational design of new active ingredients and new chemical entities. The first works using the (QSAR) method began in the early 1960s with Hansch on the one hand and Free and Wilson on the other hand, who proposed a mathematical model to correlate biological activity and chemical structure. The developments of this study (QSAR) are very old, from 1868, when Alexander Crum-Brown and Thomas Fraser show the existence of a relationship between physiological activity and chemical structure. Followed, with Richet establishing a relationship between toxicity and physicochemical properties. Independently, Meyer and Overton described a linear correlation between lipophilicity (oil-water partition coefficient) and biological (narcotic) effects. During the last decades, this field has been widely studied and the bibliographic data available on this approach are now important.

5.3 Objectives development of QSAR:

There are two main objectives for the development of QSAR:

- Development of predictive and robust QSAR, with a specified chemical domain, for prediction of activity of untested molecules.
- It acts as an informative tool by extracting significant patterns in descriptors related to the measured biological activity leading to understanding of

mechanisms of given biological activity. This could help in suggesting design of novel molecules with improved activity profile.

5.4 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.

5.5 QSAR modeling:

To obtain a good-quality QSAR model with the ability to predict activity of a chemical outside the training set depends on many factors in the approach and execution of each of the three steps:

1. Collect or if possible, design a training set of chemicals.
2. Choose descriptors that can properly relate chemical structure to biological activity.
3. Apply statistical methods that correlate changes in structure with changes in biological activity.

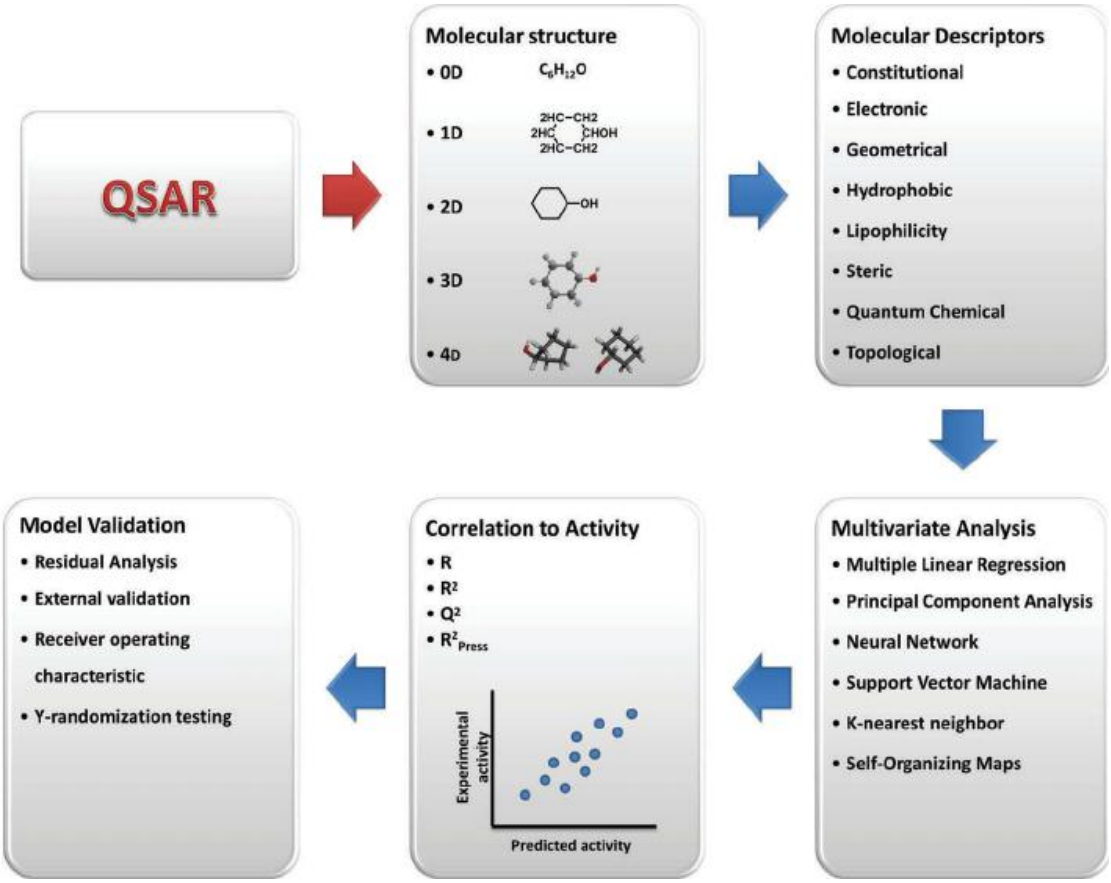


Figure 4: General-Steps-involved-in-QSAR-modeling.

5.6 QSAR Development process:

There are a lot of software available for QSAR development and they are either commercial or free. These include specialized software for drawing chemical structures, interconverting chemical file formats, generating 3D structures, calculating chemical descriptors, developing QSAR models, and general-purpose software that have all the necessary components for QSAR development. For Structure Drawing or File Conversion the most common programs are ChemDraw, ACD/ChemSketch and Open Babel software. Soft wares for 3D Structure Generation are CORINA, Concord, Frog, smi23d. Descriptor Calculation can be made by using Dragon, Molconn-Z, and PaDEL-Descriptor software.

5.7 Our QSAR Analysis can help you:

1. Establish a quantitative relationship between the chemical structure and biological activity of the compound.
2. Predict the activity of the new compound based on the structural data and speculate the possible mechanism of action.
3. Optimize the structure of existing compounds to enhance their biological activity.
4. Design novel candidate lead compounds.
5. Predict the absorption, distribution, metabolism, excretion, toxicity, and other properties of compounds.

6. QSAR CHALLENGE

Over the past decades, Quantitative Structure-Activity Relationships (QSARs) have become a powerful theoretical tool, alternative to quantum mechanics, for the description and prediction of the properties of complex molecular systems in different environments. The QSAR approach proceeds from the assumption of a one-to-one correspondence between any physical property, chemical affinity, or biological activity of a chemical compound and its molecular structure. The latter can be represented by the chemical composition, the connectivity of atoms, the potential energy surface, and the electronic wave function of a compound. Different physico-chemical molecular descriptors reflecting structure can be determined empirically or using theoretical and computational methods of different complexity. It should be emphasized that knowledge of the exact chemical constitution and/or the three-dimensional molecular structure of the chemical compounds studied is a prerequisite for the application of the QSAR approach. The success of the QSAR approach critically depends on the precise definition and appropriate use of molecular descriptors.

Quantitative structure-activity relationship (QSAR) modeling pertains to the construction of predictive models of biological activities as a function of structural and molecular information of a compound library. The concept of QSAR has typically been used for drug discovery and development and has gained wide

applicability for correlating molecular information with not only biological activities but also with other physicochemical properties.

7. Advantages and Disadvantages of QSAR:

- Advantages of predicting biological activity with quantitative structure-activity relationships modelling include:
  6. Able to predict activities of a large number of compounds with little to no prior experimental data on activity.
  7. 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.

  8. In vivo and in vitro experimentation can be very expensive and time-consuming.
  9. QSAR modelling reduces the need for testing on animals and/or on cell cultures and saves time.

- Disadvantages of predicting biological activity with QSAR modelling include:
  1. Does not provide an in-depth insight on the mechanism of biological action.
  2. Some risk of highly inaccurate predictions of pharmacological or biological activity.

7. Deep Learning

7.1. Machine Learning (ML):

Machine Learning is a method for obtaining Artificial Intelligence. It is a subset of artificial intelligence (AI) in the field of computer science that often employs statistical techniques to allow computers to "learn" ML comes in a variety of forms:

Supervised learning: Systems that are exposed to vast volumes of labeled data are used. To master a mission, some systems may require exposure to millions of examples (Alpaydin, 2020).

Unsupervised learning: Refers to programs that attempt to spot trends in data to divide them into categories (Alpaydin, 2020).

Reinforcement (Semi-supervised) learning: Combines supervised and unsupervised learning to train systems with small amounts of labeled and large amounts of unlabeled data. The labeled data is used to partially train a machine-learning model, which is then used to mark the unlabeled data in a method known as pseudo-labeling (Alpaydin, 2020).

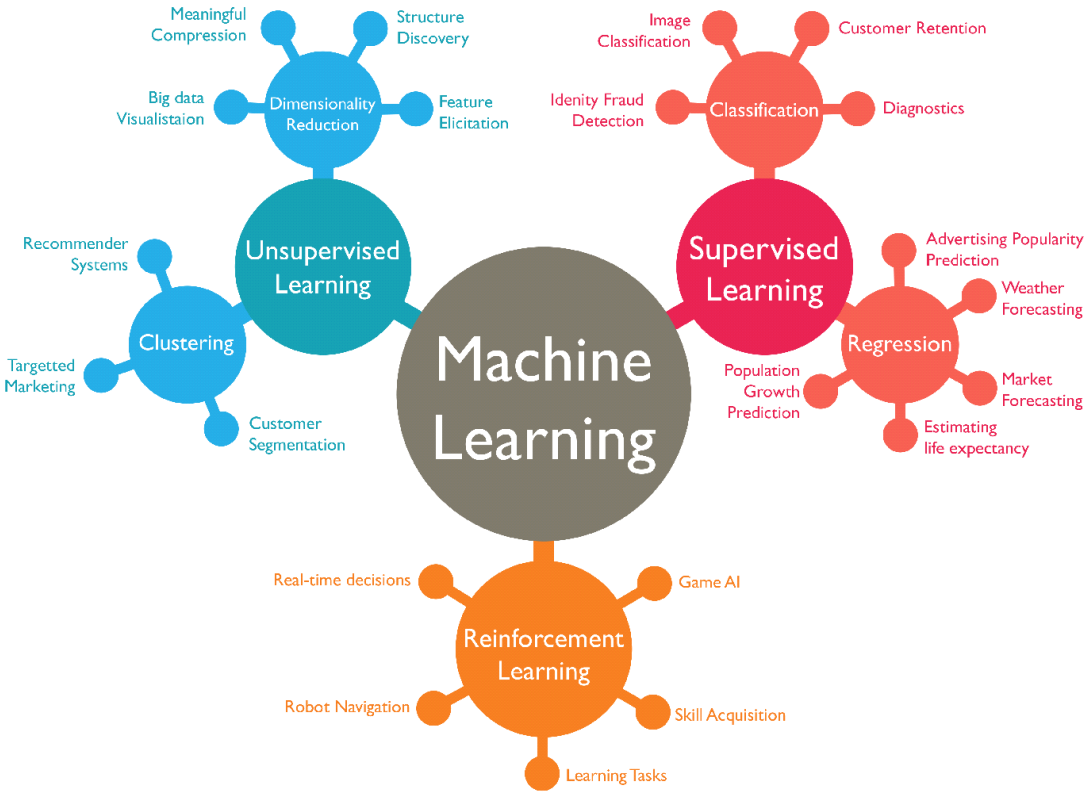


Figure 5: Types of Machine leaning.

7.2. Definition Deep Learning (DL):

Deep Learning is a machine learning implementation technique. It is a subset of machine learning that has networks capable of learning unsupervised from unstructured or unlabeled data. Large amounts of unstructured data, such as human voice, text, and images, are processed by DL. A deep learning model is a type of artificial neural network that consists of multiple layers of mathematical computations on data, with the results from one layer being fed as inputs into the next layer to

classify and/or predict the input data. DL algorithms are taught to accomplish the same tasks for machines as we use our brains to recognize patterns and classify different types of knowledge. Deep learning, in comparison to machine learning, requires high_end computers and a large amount of training data to produce accurate results (Yoshua Bengio et al., 2003).

Deep Learning (DL) processes data, understands human expression, and visually recognizes objects through layering algorithms. Each layer passes information to the next, with the previous layer's output serving as input to the next. The input layer is the first layer in a network, while the output layer is the last. Hidden layers refer to the layers that exist between the two. Each layer is usually a straightforward, uniform algorithm with only one type of activation function (Schmidhuber, 2015).

What Is the Difference between Deep Learning and Machine Learning?

Machine Learning (ML) is basically used to implement Artificial Intelligence. Machine Learning works efficiently when we have some moderate amount of sorted data which we train and get the desired results according to the problem we need to solve. Machine Learning working does not depend on the type of system or hardware we are using, it works efficiently on every system.

Machine Learning takes less time to train data and get ready for analyzing results. It works by converting the complexity of data to a lower level so that analysis of results get more efficiency and produce more curate result. It starts by matching pattern by doing so it breaks a problem into sub_problems as it is easy to get an output from a short problem then at the end it cumulate the results to get the final result as in other words.

Table 4: Diversity between machine learning (ML) and deep learning (DL).

Deep Learning (DL):	Machine Learning (ML)
Requires a lot of unlabeled training data to make concise conclusions .	Can use small data amounts provided by users.
Needs high-performance hardware.	Not needs high-performance hardware.
Creates new features by itself.	Requires features to be accurately identified by users
Solves the problem on the end-to-end basis.	Divides tasks into small pieces and then combine received results into one conclusion
Needs much more time to train.	Not needs much more time to train.
Can not provide enough transparency for its decisions.	Can provide enough transparency for its decisions.

The concept of deep learning implies that the machine creates its functionality by itself as long as it is possible at the current time. To infer, deep learning applications use a hierarchical approach involving determining the most important characteristics to compare.

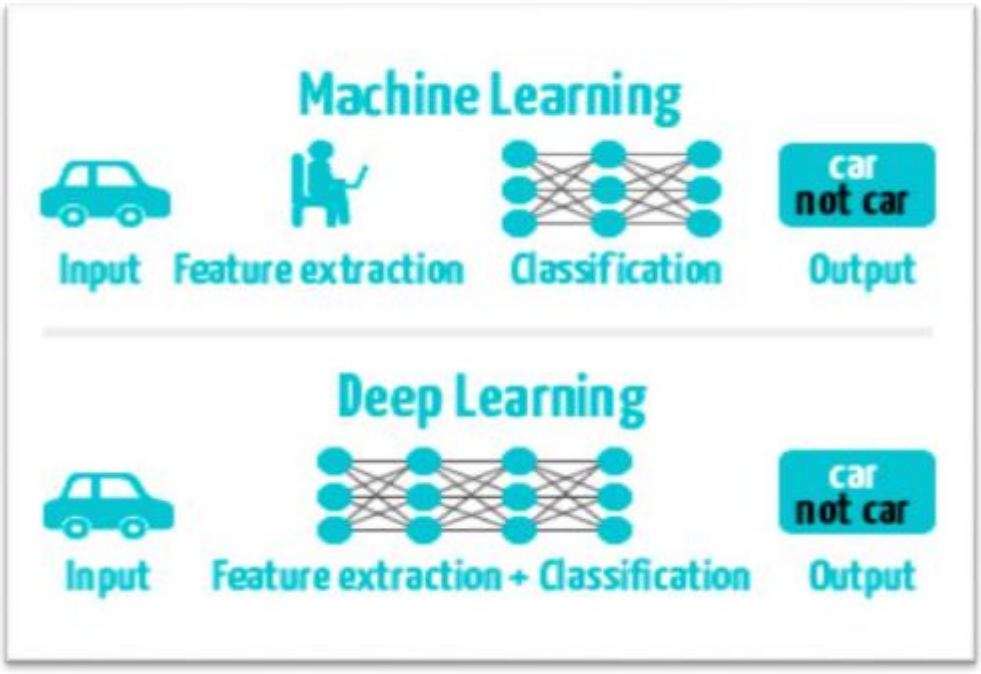


Figure 6: Difference between Machine Learning and Deep Learning.

7.1 Convolutional neural network (CNN)

7.1.1 Convolutional neural network Definition:

Convolutional neural network (CNN) is a type of deep learning model for processing data that has a grid pattern, such as images, and designed to automatically and adaptively learn spatial hierarchies of features, from low- to high-level patterns.

A Convolutional neural network (CNN) is a neural network that has one or more convolutional layers and are used mainly to provide a higher level of efficiency and accuracy in all the fields in which it has been employed in most popular of which are Object Detection, Digit Recognition and image processing and classification and to segment and also for other auto correlated data. A convolution is essentially sliding a filter over the input.

7.1.2 CNN Architecture (5 Layers):

The CNN architecture consists of several kinds of layers; Convolutional layer, pooling layer, fully connected input layer, fully connected layer and fully connected output layer.

- **Convolutional layer:** Convolutional layer is the backbone of any CNN working model. This layer is the one where pixel by pixel scanning takes place of the images and creates a feature map to define future classifications.
- **Pooling layer**: Pooling is also known as the down-sampling of the data by bringing the overall dimensions of the images. The information of each feature from each convolutional layer is limited down to only containing the most necessary data. The process of creating convolutional layers and applying pooling is continuous, may take several times.
- **Fully connected input layer:** This is also known as the flattening of the images. The outputs gained from the last layer are flattened into a single vector so that it can be used as the input data from the upcoming layer.
- **Fully connected layer:** After the feature analysis has been done and it's time for computation, this layer assigns random weights to the inputs and predicts a suitable label.

- **Fully connected Output layer:** This is the final layer of the CNN model which contains the results of the labels determined for the classification and assigns a class to the images.

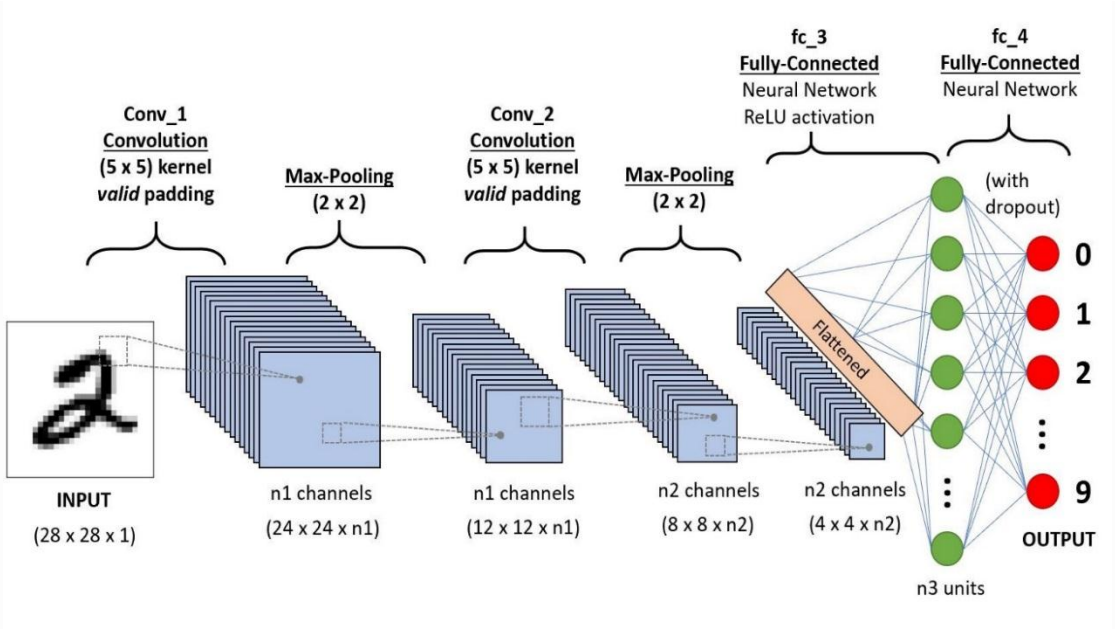


Figure 7: Example of A CNN sequence to classify handwritten digits.

7.1.3 Advantages and disadvantages of CNN

The advantage of CNN:

1. The main reason to consider CNN is the weight sharing feature, which reduces the number of trainable network parameters
2. Concurrently learning the feature extraction layers and the classification layer causes the model output to be both highly organized and highly reliant on the extracted features.
3. Large-scale network implementation is much easier with CNN than with other neural networks.

The disadvantages of CNN models are:

1. A Convolutional neural network is significantly slower due to an operation such as maxpool.
2. If the CNN has several layers then the training process takes a lot of time if the computer doesn't consist of a good GPU.
3. A ConvNet requires a large Dataset to process and train the neural network.

7.2 Graph Neural Network

7.2.1 Graph Neural Network definition:

Graph Neural Networks (GNNs) are a class of deep learning methods designed to perform inference on data described by graphs, which are a very active, new field of research that has a tremendous potential, because there are many datasets in real-life applications that can be structured as graphs. A graph can represent things like social media networks, or molecules.

Graph neural networks are proposed to combine the feature information and the graph structure to learn better representations on graphs via feature propagation and aggregation. Due to its convincing performance and high interpretability, GNN has recently become a widely applied graph analysis too. GNNs can do what Convolutional Neural Networks (CNNs) failed to do.

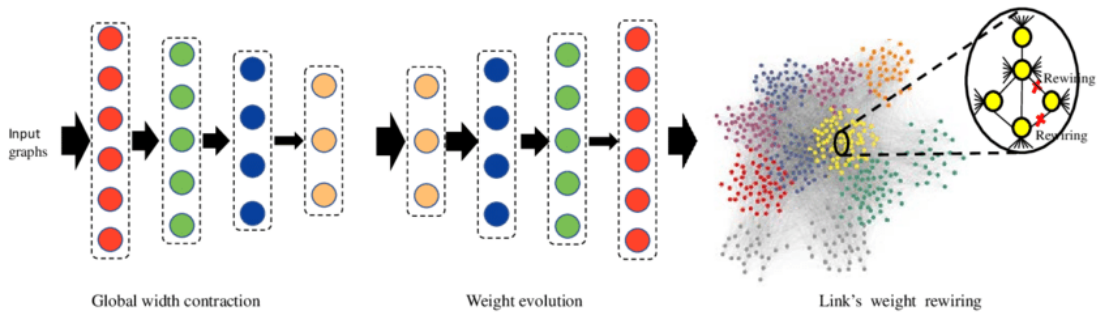


Figure 8: A graph neural network architecture design.

7.2.2 Applications of GNNs:

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

- Node Classification
- Link Prédiction
- Graph Classification
- Graph visualization
- Graph clustering

Node Classification: the task here is to determine the labeling of samples (represented as nodes) by looking at the labels of their neighbors. Usually, problems of this type are trained in a semi-supervised way, with only a part of the graph being labeled.

Link prediction: here, the algorithm has to understand the relationship between entities in graphs, and it also tries to predict whether there's a connection between two entities. It's essential in social networks to infer social interactions or to suggest possible friends to the users. It has also been used in recommender system problems and in predicting criminal associations.

Graph Classification: the task here is to classify the whole graph into different categories. It's like image classification, but the target changes into the graph domain. The applications of graph classification are numerous and range from determining whether a protein is an enzyme or not in bioinformatics, to categorizing documents in NLP, or social network analysis.

Graph visualization: is an area of mathematics and computer science, at the intersection of geometric graph theory and information visualization. It is concerned with the visual representation of graphs that reveals structures and anomalies that may be present in the data and helps the user to understand the graphs.

Graph clustering: refers to the clustering of data in the form of graphs. There are two distinct forms of clustering performed on graph data. Vertex clustering seeks to cluster the nodes of the graph into groups of densely connected regions based on either edge weights or edge distances. The second form of graph clustering treats the graphs as the objects to be clustered and clusters these objects based on similarity.

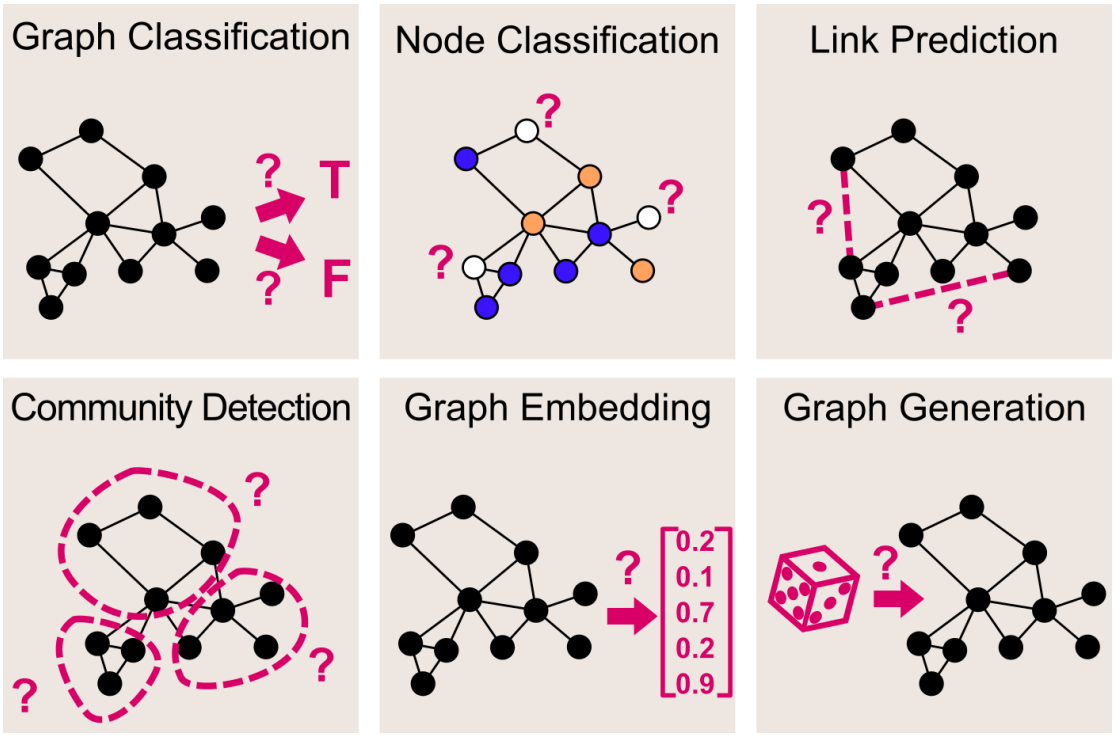


Figure 9: Difference between graph and node classification.

7.2.3 Types of graph neural networks (GNN):

Although a basic approach of a Graph Neural Network is an effective method of analysis, it may provide limitation to the desired field of research. A solution to such issue is the use of different GNN types, according to the type of graph or data required. Such GNN types are:

1. Graph Convolutional Network
2. Graph Attention Network
3. Gated Graph Neural Network

Recurrent Graph Neural Network

RecGNN is based on the Banach Fixed-Point Theorem, which asserts that: Let (X, d) be a full metric space and (T: X→X) be a contraction mapping. Then T has a unique fixed point (x∗), and for each x∈X, the sequence T n(x) for n→∞ converges to (x∗). This suggests that if I apply the mapping T to x for k times, x^k should be close to X^ (k-1).

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

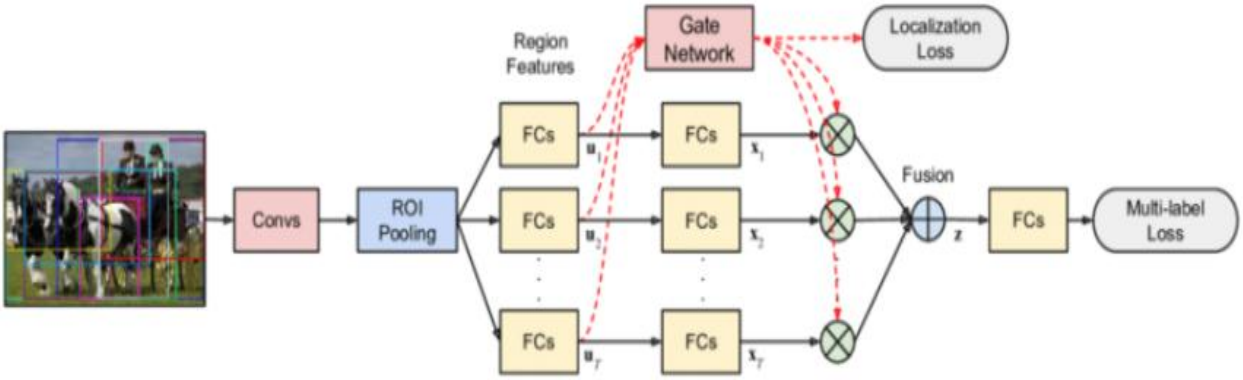


Figure 10: Architecture of Recurrent Graph Neural Network.

Spatial Convolutional Network

Spatial Convolutional Networks have a similar idea to CNNs. As is well known in CNN, convolution is performed by summing the neighboring pixels around a central pixel using a filter and learnable weights. Spatial Convolutional Networks operate on a similar principle, aggregating the properties of neighboring nodes toward the center node.

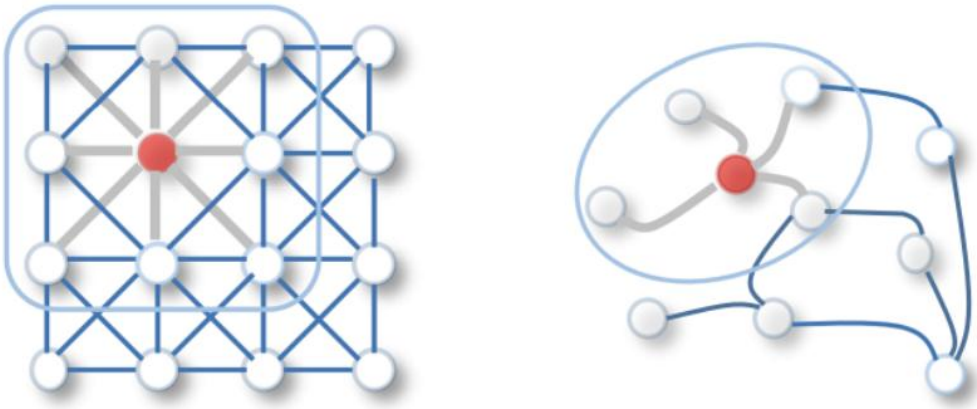


Figure 11: Image a Comprehensive Survey on Graph Neural Networks Left: Convolution on a regular graph such as an image. Right: Convolution on the arbitrary graph structure.

Spectral Convolutional Network

In comparison to other types of Graph Neural Networks, this sort of GNN is built on a solid mathematical foundation. It is based on the theory of Graph Signal Processing. It simplifies by the use of Chebyshev polynomial approximation.

7.3. Graph convolutional neural networks

7.3.1 Definition of graph convolutional neural networks:

GCNs are a very powerful neural network architecture for machine learning on graphs. In fact, they are so powerful that even *a randomly initiated 2-layer GCN* can produce useful feature representations of nodes in networks. The figure below illustrates a 2-dimensional representation of each node in a network produced by such a GCN. Notice that the relative nearness of nodes in the network is preserved in the 2-dimensional representation even without any training.

ConvGNNs are closely related to recurrent graph neural networks. Instead of iterating node states with contractive constraints.

7.3.2 Architecture of the convolutional neural network on graphs

The present study proposes a learning model on graph-structured data for the classification of building patterns by combining the graph convolution with the computational mode of the neural network. The model consists of multiple convolutional and pooling layers and one fully connected layer. The architecture of the model with two convolutional layers is shown.

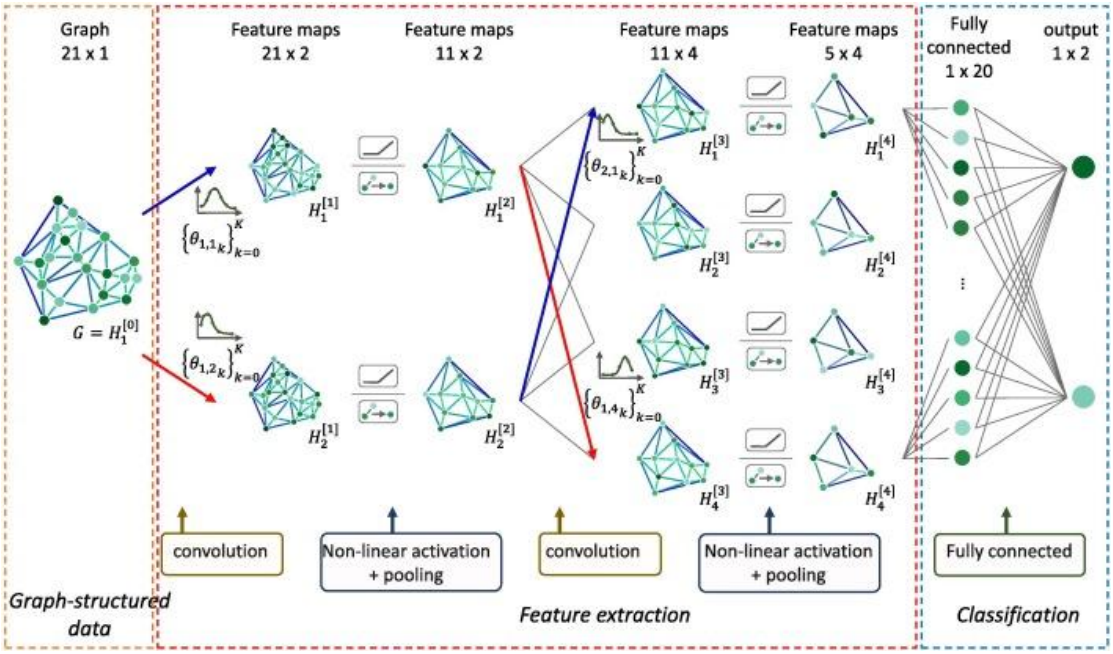


Figure 12: Architecture of the convolutional neural network

7.3.3 Application of convolutional neural networks to graphs

- Graph Convolutional Networks generate predictions over physical systems, such as graphs, their interactive approach and applications. GCN also provides accurate information about the properties of real-world entities and physical systems (dynamics of the collision, objects trajectories).
- GCNs are used to perform image differentiation problems. The model it follows is known as 'Zero-Shot Learning'. The main motive of this model is to identify an unknown labelled image and group it into known ones. They also gather semantic information of these labels and categorise them.
- GCNs can take a certain length of molecular fingerprints as input and generate predicted molecular structures. MolGAN is one kind of Graph Convolutional network which helps to create new molecular structures with various features in it. In this way, it allows scientists to invent modern molecular structures day by day.
- GCN is applicable for solving various problems related to research operations and combinatorial optimisation applications. Graph Convolutional Networks play a pivotal role in solving salesman problems, quadratic assignment problems, and many more. With the help of the input graph, it can outclass traditional complex algorithms.

8. Discussion:

As we have seen, neural networks were widely used for quantitative structure–activity relationships (QSAR). Because of various practical issues (e.g., slow on large problems, difficult to train, prone to over fitting, etc.) and for this problem We proposed a new type of individual QSAR classifier that is deep neural network model based on graphs neural networks (GNNs), It automatically extracts sequential features from a simplified molecular-input line-entry system (fingerprints),along with her ability of to reveal complex and nonlinear relationships, have led to sophisticated performance model effective for similarity methods and QSAR measurement. And analyzing those data in the context of graph neural networks (GNNs).

GNNs is from the best models used deep learning for biological activity prediction molecular , a model which builds a set of learning models and combines multiple models to produce final and effective predictions. Theoretically and empirically, it has been shown that the predictive power of ensemble learning surpasses that of a single individual learner if the individual algorithms are accurate and diverse.

As we have seen, there are many different types of molecular descriptors which are typically designed to provide a molecular description in chemical space. Thus, it is important to select descriptors that are the most appropriate for an application. Some of the descriptors are not suitable for applications like diversity, Studies such as those derived from quantum mechanics which are considered computationally expensive. The ideal descriptor must be able to discriminate between different molecules in terms of their bioactivity, physicochemical or any other properties. Usually, two molecules may be absolutely similar with respect to one property (e.g. molecular weight) and completely different with respect to another property (e.g. hydrophobicity). Therefore, it becomes clear why several hundreds of descriptors are used in chemical information systems. Typically, chemical databases used for evaluation purposes contain both structural and activity data. Selecting the best descriptors is based on their use in predicting the activity of a molecule from another molecule that is considered similar to it, either by using a certain methods similarity searching of molecular structure, clustering or using its k-nearest neighbours . For those descriptors, prediction of their activity values, compared to the actual observed values, given by either the product moment The similarity coefficient,tanimoto coefficient or weighting scheme, will be

used to judge descriptor in case of Molecular similarity with Apply information retrieval measures using Tanimoto coefficient from database.

For qualitative activities, like predicting the activity class of molecules, the best descriptors are those yielding the highest number of correct predictions (molecules with similar activity class), taking into account the total number of molecules having that activity class in the database used. Another way to evaluate the descriptors is by looking at the similarity-activity correlation, with the higher correlation indicating a better measure in terms of the similarity property principle.

A lot of studies found the Tanimoto coefficient is the most suitable measure to identify structures that are closely related in activity. Many other studies have investigated the performance of several similarity coefficients, showing that the Tanimoto coefficient is superior to other coefficients. Currently, the Tanimoto coefficient has been the most frequently used in chemoinformatics and has become the standard for measuring the structural similarity of compounds, another reason for its popularity is that the Tanimoto coefficient calculation is very fast, because it does not involve a square root.

9. Summary:

In this chapter, we have provided an overview of current machine learning methods. Through this chapter the concept of deep learning methods was developed, and problems the challenges indicated by molecular similarity through by graph neural networks presented of quantitative structure relationship (QSAR) method.

Chapter 3:

Research methodology

Table of content:

8. Introduction……43
9. Research Framework……43
9.1 Research plan and review of literature……45
9.2 Collection of chemical dataset……45
9.3 Converting molecule data from molecule descriptors……46
10. Ensemble learning……49
3.1 application of information retrieval similarity using tanimonto……49
11. Design and steps……51
11.1 Global Architecture……51
11.1.4 Find graph structure……53
11.1.5 Specify graph type and scale……53
11.1.6 Design loss function……54
11.1.7 Build model using computational modules……54

8 Performance Evaluation……...……55
9 Benchmarking……...…56
10 Summary……..57

1. Introduction:

At present, QSAR methodologies comprise a series of possibilities, including joining forces with machine learning methods and increasing even more the swiftness they might bring to the prospective development of therapeutics in the Health Sciences scope.

Developing accurate and interpretable QSAR models is therefore of the utmost importance in the drug discovery process. Deep neural networks that are powerful supervised learning algorithms, have shown great promise for addressing regression and classification problems in various research fields, including the pharmaceutical industry.

In this chapter, we present our proposed solution to deal with this problem, based on deep learning, which aim of this research is to develop a novel method of biological activity prediction, based on deep learning Graph neural network (GNN) and QSAR methodologies, In order to enhance the effectiveness of the similarity methods and QSAR measurement.

1. Research Framework:

The study methods will be conducted according to the workflow process as illustrated in Figure 13.

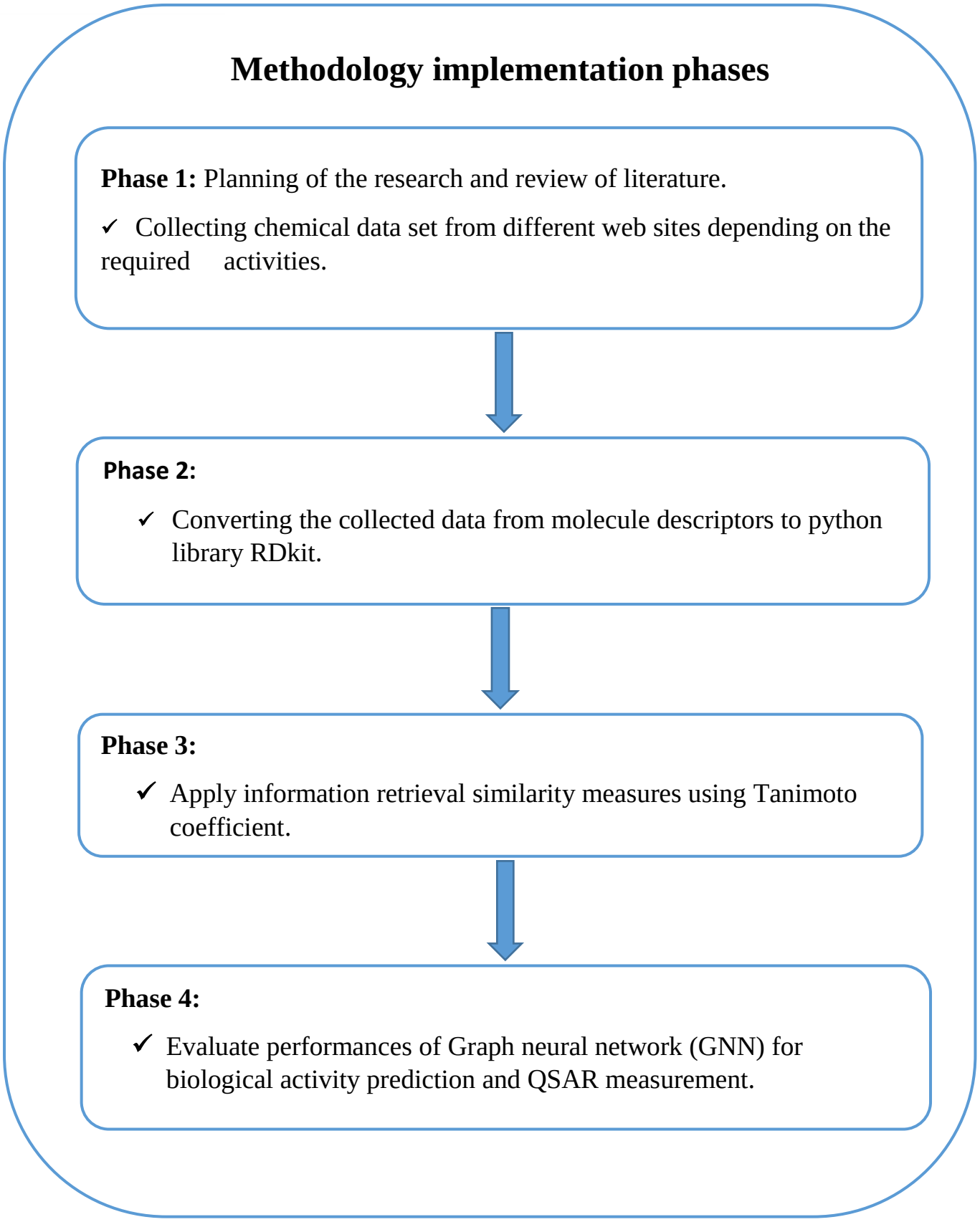


Figure 13: Flow Chart of the Framework.

2.1 Research Plan and Review of Literature:

The work plan include understanding the representation of chemical structures and retrieval of data from chemical structure, molecular descriptors for similarity searching and using similarity coefficients to get similarities between the molecules and predict biological activity , Chemical database containing 8294 compounds have been used. The main requirement in this research is to develop a novel method of biological activity prediction, based on deep learning Graph neural network (GNN).

2.2 Collection of Chemical Data Set:

Chemical databases containing huge number of compounds are available in many web sites. In this study, the required data is MDL Drug Database Report (MDDR) (MDDR is a database covering the patent literature, journals, meetings and congresses). Produced by Molecular Design Limited's (MDL) and Prous Science, the database contains over 100,000 compound biologically relevant compounds and well-defined derivatives, with updates adding about 10,000 a year to the database.

The MDDR Finder allows searching the database by structure or across relevant data fields. MDL also offers MDDR-3D collected from the Discovery Gate web site (URL:https://www.discoverygate.com).

Although the MDDR (MDL Drug Database Report) contains many chemical compounds in groups of activities with different biological similarities, the data collection is focused on selecting some compounds of different activities.

MDDR collection of datasets remains among the most common databases used in chemo-informatics. It which include several hundred diverse activities, we experiment used dataset from the common MDDR database, which are denoted as MDDR-DS1.

The data represented here composed from active compounds with different of the activity for each of them. There are two data's as shown in figure 3.3;

the first data has seven actives. The second has three actives, the first compound from each part selected as active target (query), all other compounds are assumed to be inactive, to find the similarity to that target. The process will continue the some way for other parts.

Table 5: MDDR activity classes for DS1 data set.

Activity index	Activity class	Active molecules	Pairwise similarity
31420	Renin inhibitors	1130	0.290
71523	HIV protease inhibitors	750	0.198
37110	Thrombin inhibitors	803	0.180
31432	Angiotensin II AT1 antagonists	943	0.229
42731	Substance P antagonists	1246	0.149
06233	5HT3 antagonist	752	0.140
06245	5HT reuptake inhibitors	359	0.122
07701	D2 antagonists	395	0.138
06235	5HT1A agonists	827	0.133
78374	Protein kinase C inhibitors	453	0.120
78331	Cyclooxygenase inhibitors	636	0.108

2.3 Converting Molecule data from molecule descriptors with python library RDKit:

Data Representation

The input data for DL-based drug discovery are molecules; drugs and protein targets are small molecules or macromolecules. To characterize these molecules, several types of molecular representations (often referred to as descriptors or features) have been used in many machine learning (ML) methods—from simple sequences of molecular entities to manually predefined molecular features .However, because it is directly related to the knowledge of learning models, data representation has a significant impact on pre-training to improve performance of predictive models.

SMILES to graph

Molecular string representations known as SMILES were used for encoding the molecular descrepteur. Then, for each molecule the cheminformatics package RDKit (version 2021.03.1) was used to calculate atom and bond features to be encoded into a graph. This graph G = (V, E) is defined by a set of nodes V and edges E representing the corresponding atoms and chemical bonds. Each node and edge has a vector of features containing relevant structural information of the molecule. The features used in this work are summarized in Tables 1 and 2 for nodes and edges, respectively. These features were chosen with the aim of distinguishing fundamental differences among atoms and chemical bonds present within the same molecule and were inspired by previous works using similar schemes.

The dimension of each feature corresponds to the number of different classes available within the studied dataset for such feature. The exception to this are the features which contained only two different classes (i.e., Ring and Aromatic for node features (Table 1), and Conjugated and Ring for edge features (Table 2)) which are encoded as a single binary value. In this way, each node/edge feature is represented as a vector of the corresponding dimension indicating the presence (value 1) or absence (value 0) of each class in the feature. where na and nb refer to the number of atoms and bonds in the molecule, respectively, and the numbers 25 and 6 correspond to the dimensions (as a result of the concatenation of all feature vectors) of each node and edge representation, respectively. Besides the feature vectors, the connectivity information of the graph has to be also defined. At the end, the SMILES of molecules are transformed into graphs representing the corresponding species. In order to have smaller graphs, hydrogen atoms are explicitly encoded (see Table 6 for Attached Hs) into the node features vector of heavier atoms.

To translate the text-form structure (i.e., SMILES string) into a DL-acceptable type (i.e. molecular graphs, images), the molecular descriptors were created by a feature extraction algorithm to characterize the molecular structure. Specifically, the default node (atom) and edge (bond) representations are based on Weave Net paper.

At the end, the SMILES of the dataset are transformed into graphs representing the corresponding species. In order to have smaller graphs, hydrogen atoms are explicitly encoded (see Table 7 for Attached Hs) into the node features vector of heavier atoms.

Table 6: Atom features used to define the initial nodes in the molecular graphs.

Feature	Description	Dimension
Atom type	(C, Br, Cl, N, O, I, S, F, P)	9
Ring	Is the atom in a ring?	1
Aromatic	Is the atom part of an aromatic system?	1
Hybridization	(sp, sp^2, sp^3)	3
Bonds	Number of bonds the atom is involved in	4
Charge	Atom's formal charge	3
Attached Hs	Number of bonded hydrogen atoms	4

Table 7: Bond features used to define the initial edges in the molecular graphs.

Feature	Description	Dimension
Bond type	(Single, double, triple, aromatic)	4
Conjugated	Whether the bond is conjugated	1
Ring	Whether the bond is part of a ring	1

Then, RDKit is used to extract the atom features, bond features and connectivity. Then, Pytorch Geometric (PyG)49 is employed to embed the corresponding matrices A and B into the graph with the connectivity described by matrix C.

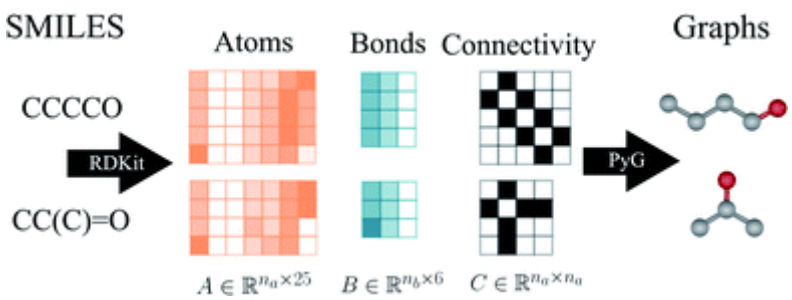


Figure 14: Schematic illustration of the construction of initial graphs from the SMILES of dataset.

3. Ensemble Learning:

At this stage, data is prepared to be fed to a model training procedure. The model training procedure will consist of performing a search for the best hyper-parameters for each type of model that is intended to be trained. Out of this hyper-parameter search, the best model of each type for the current dataset will be produced as an output. The hyper-parameter search is executed by a well-known procedure called cross-validation, where the dataset is split into training, validation, and testing. The model is trained on the training split and tested on the validation split until the best set of parameters is found. Then finally, to assess the generalization power of the best model, it is tested again with the test data split. The more advanced way to perform the validation is to split only training and testing, and divide the training split into k folds, where k-1 folds are used for training and the remaining fold is used for validation. The average metric score on all the training-validation fold permutations is used to select the best set of parameters. The two tasks undertaken in this thesis belong to node regression and graph classification. For the node regression, since we want to approximate a real value for nodes of a graph, the normalized root means the squared error is the correct performance metric to use. However, the problem will be transformed into a classification task by defining between the query and the molecular database. thus the tanimoto coefficient used to obtain a numeric quantification to the degree of similarity between a pair of structures, is the right performance metric to use.

3.1 Application of information retrieval similarity measures using Tanimoto coefficient:

Retrieval of Data from Chemical Database

Retrieval of data from chemical structure search system can offer three principle types of searching facility of retrieval mechanisms these are structure, substructure and similarity searching. This expands the capabilities of the existing systems by capitalizing on the strengths of relational database technology. The system allows the user to optimally store and search chemical structure information including information related to multi-valued atoms and multi-typed bonds, Several similarity methods have been developed in deep learning to calculate the similarity between the

query and the molecular database. In this study, the continuous Tanimoto measure we utilized to calculate the similarity of the molecules in the represented datasets, the retrieval of similarity coefficients is used to obtain a numeric quantification to the degree of similarity between a pair of structures.

Tanimoto coefficient is a metric (or score) to measure the similarity of two sets of elements, which is uses to calculate the degree of similarity between the query and the target structures. The Tanimoto coefficient has two arguments:

• The fingerprint of the query structure

• The fingerprint of the target structure

A fingerprint is comprised of a list of predefined structure fragments or feature found within a structure, which is a bit string (a sequence of "0" and "1" digits) that contains information on the structure. 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, with formula:

$$T = NA\&B / NA + NB - NA\&B$$

NA represents the number of "on" features (bits) in structure A.

NB represents the number of "on" features (bits) in structure B.

NA&B represents the number of "on" features (bits) common to both fingerprints A and B.

The process of fingerprint generation is as follows:

1. Up to a given bond, all linear paths (linear patterns) consisting of bonds and atoms of a structure are detected.
2. Branching points at the end of each linear pattern are also detected.
3. All cycle (cyclic patterns) are detected.
4. Using a proprietary hashing method, a given number of bits in the bit stream are set for each pattern. It is possible that the same bit is set by multiple patterns. This phenomenon is called bit collision. A few bit collisions in the

fingerprint are tolerable, but too many may result in losing information in the fingerprint.

Below is an example how calculate The Tanimoto Similarity Measures as shown in table 8.

Table 8: 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

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

5. Design and steps

5.1 Global Architecture:

Underlying process of creating and using a QSAR model. Molecules are collected and descriptors are calculated that the model is created using the Training Set and the model is then tested on the Test Set.

Here, we present the different phases followed to design our proposed approach, furthermore explaining the details of each phase, starting by giving a general diagram on our approach ,which goal is trying to reach an acceptable biological activity prediction for molecules with deep learning we focus on the data set MDDR to be rich.

In the other hand, using graph neural network to enhance our model to see if it will improve the results of biological activity prediction of the molecule.

We organize our system according to a single categorization approach, which is the deep learning approach, and use graph neural network to categorize molecules in the test corpus.

The basic idea behind most GNN architectures is graph convolution. In essence, we try to generalize the idea of convolution into variants of graph neural networks (GNNs) are proposed to collectively aggregate information from graph structure. Thus, they can model input and/or output consisting of elements and their dependency.

We present bellow the general architecture .giving the possible steps:

- Clean and prepare data.
- Develop and training GNN Model.
- Testing and evaluation model skill.

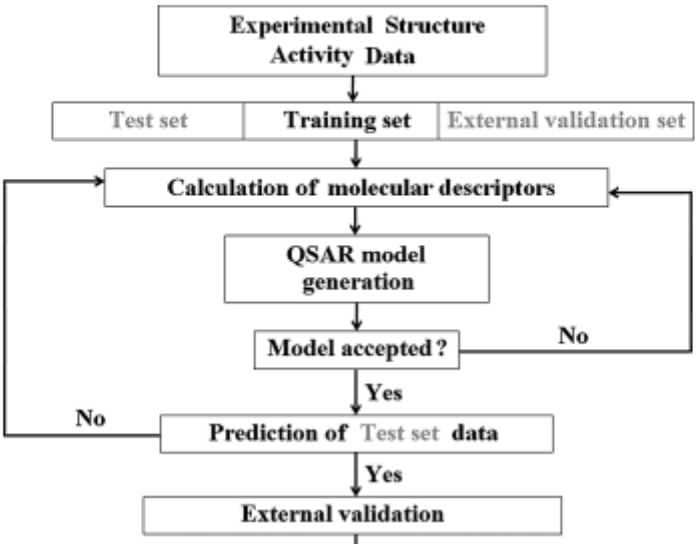


Figure 15: global architecture of QSAR model.

- In this section, we present the general design pipeline of a GNN model for a specific task on a specific graph type. Generally, the pipeline contains four steps:

1) Find graph structure.
2) Specify graph type and scale.
3) Design loss function.
4) Build model using computational modules.

We give general design principles and some background knowledge in this section. The design details of these steps are discussed in later sections.

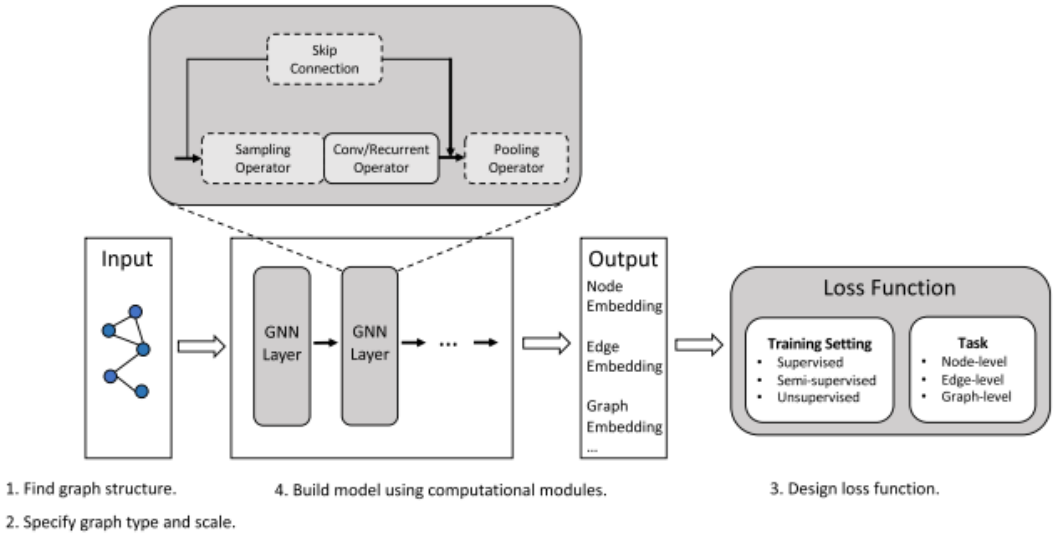


Figure 16: The general design pipeline for a GNN model.

5.2 Find graph structure

At first, we have to find out the graph structure in the application. There are usually two scenarios: structural scenarios and non-structural scenarios. In structural scenarios, the graph structure is explicit in the applications, such as applications on molecules, physical systems, knowledge graphs and so on. In non-structural scenarios, graphs are implicit so that we have to first build the graph from the task, After we get the graph, the later design process attempts to find an optimal GNN model on this specific graph.

5.3 Specify graph type and scale

After we get the graph in the application, we then have to find out the graph type and its scale.

Graphs with complex types could provide more information on nodes and their connections. Graphs are usually categorized as:

• **Directed/Undirected Graphs.** Edges in directed graphs are all directed from one node to another, which provide more information than undirected graphs. Each edge in undirected graphs can also be regarded as two directed edges.

• **Homogeneous/Heterogeneous Graphs.** Nodes and edges in homogeneous graphs have same types, while nodes and edges have different types in heterogeneous graphs. Types for nodes and edges play important roles in heterogeneous graphs and should be further considered.

• **Static/Dynamic Graphs.** When input features or the topology of the graph vary with time, the graph is regarded as a dynamic graph. The time information should be carefully considered in dynamic graphs.

5.4 Design loss function

In this step, we should design the loss function based on our task type and the training setting. For graph learning tasks, there are usually three kinds of tasks:

•**Node-level** tasks focus on nodes, which include node classification, node regression, node clustering, etc. Node classification tries to categorize nodes into several classes, and node regression predicts a continuous value for each node. Node clustering aims to partition the nodes into several disjoint groups, where similar nodes should be in the same group.

•**Edge-level** tasks are edge classification and link prediction, which require the model to classify edge types or predict whether there is an edge existing between two given nodes.

•**Graph-level** tasks include graph classification, graph regression, and graph matching, all of which need the model to learn graph representations.

5.7 Build model using computational modules

Finally, we can start building the model using the computational modules. Some commonly used computational modules are:

• **Propagation Module.** The propagation module is used to propagate information between nodes so that the aggregated information could capture both feature and topological information. In propagation modules, the **convolution operator** and **recurrent operator** are usually used to aggregate information from neighbors while the **skip connection** operation is used to gather information from historical representations of nodes and mitigate the over-smoothing problem.

• **Sampling Module.** When graphs are large, sampling modules are usually needed to conduct propagation on graphs. The sampling module is usually combined with the propagation module.

• **Pooling Module.** When we need the representations of high-level subgraphs or graphs, pooling modules are needed to extract information from nodes.

With these computation modules, a typical GNN model is usually built by combining them. A typical architecture of the GNN model is illustrated in the middle part of Figure. 2 where the convolutional operator, recurrent operator, sampling module and skip connection are used to propagate information in each layer and then the pooling module is added to extract high-level information. These layers are usually stacked to obtain better representations.

6. Performance of QSAR:

The performance of the QSAR model for the MDDR datasets was estimated by graph convolutional network (GNN).

Graph neural networks (GNNs) have been demonstrated to be an effective model for learning tasks related to graph structured data. Different from classical deep neural networks which handle relatively small individual samples, GNNs process very large graphs, which must be partitioned and processed in a distributed manner. We present GNN approach for biological activity prediction and QSAR measurement.

In this study, different number of compound from either small or large data was being tried with different database parameters, through long or short queries. Here we approach the problem through deep neural networks, as they are flexible in modeling structured data such as grids, sequences and graphs.

7. Benchmarking

Predicting biological activities of molecules in the target environments is a crucial step in the drug discovery pipeline. Much research has focused on the analysis of quantitative structure-activity relationships (QSAR), which results in a myriad of molecular descriptors.

Quantitative structure−activity relationships (QSAR) is a very commonly used technique in the pharmaceutical industry for predicting on-target and off-target activities. Such predictions help prioritize the experiments during the drug discovery process and, it is hoped, will substantially reduce the experimental work

For the last 15 years, machine learning has played an important role in the prediction pipeline, which is mapping the molecular descriptors into its activity classes. Successful machine learning methods are well established, including kernel methods, random forests and gradient boosting. These models take as input a fixed-size feature vector that represents molecular properties, as known as fingerprints. The fingerprint encodes the presence of substructures in a molecule, which are then hashed into a fixed size feature vector. However, the number of substructures in large molecules might be huge, leading to many hash collisions and information loss.

Another crucial property of deep learning is that it is very flexible in modeling data structures such as images, sequences and graphs. Neural networks working directly on graphs, such as Graph Neural Network, diffusion-CNN and Column Network, can model molecular structures. Recently, there has been deep learning models successfully applied on molecular data, most models start with node representations by taking, into account of the neighborhood structures, typically through convolution and/or recurrent operations. Node representations can then be aggregated into graph representation. We conjecture that a better way is to learn graph representation directly and simultaneously with node representation.

We aim to efficiently learn distributed representation of graph, that is, a map that turns variable-size graphs into fixed size vectors or matrices. Such a representation would benefit greatly from a powerful pool of data manipulation tools. This rule out traditional approaches such as graph kernels and graph feature engineering, which could be either computation or labor intensive.

Another challenge in biological activity prediction is that the screening process is time-consuming, limiting the number of molecules to be tested. Traditional training of a small dataset with deep neural networks might cause over fitting. To improve the prediction performance, work has been proposed to use graph neural networks to graphs can also be good sources of data to train machine learning models for complicated tasks, which are a type of machine learning algorithm that can extract important information from graphs and make useful predictions.GNNs have been shown to be an effective tool for solving issues that can be represented as graphs. This is because of its adaptability, expressive capability, and ease of visualization. As a result, GNNs is an easy-to-understand solution for unstructured data that may be used in a variety of real-world settings.

This work considers strategies to develop accurate and reliable graph neural networks (GNNs) for molecular property predictions. Prediction performance of GNNs is highly sensitive to the change in various parameters due to the inherent challenges in molecular machine learning, such as a deficient amount of data samples and bias in data distribution. Comparative studies with well-designed experiments are thus important to clearly understand which GNNs are powerful for molecular supervised learning. Our work presents a number of ablation studies along with a guideline to train and utilize GNNs for both molecular regression and classification tasks.

8. Summary:

This chapter discussed the methodology, which is used in implementing this project, it has four Phases, starting from the first phase that has: planning of the research and review of literature and collections of the chemical datasets. Second phase is converting the collected data to python library RDkit. Third Phase apply information retrieval similarity measures using Tanimoto coefficient. Finally evaluating performances of Graph neural network (GNN) for biological activity prediction and QSAR measurement. GNN model helps to measure shortness. Each of these stages plays an important role in making this happen study.

Chapter 4

Implementation, Results and discussions

Table of content:

1. Introduction……58

2. Development environment……58

2.1 python……58

2.2 python features……59

2.3 python module used……61

2.4 Google colaboratory……63

2.4.1 Google colaboratory definition……63

2.4.2 benefits of colab……64

2.5 Google drive……65

3. Explanation of our approach……66

3.1 Dataset……66

3.2 Data download……66

4. Data preprocessing……67

4.1 splitting the data……67

4.2 train ,validation and test……67

4.3 network architecture……68

4.3.1 GNN implementation……68

4.4 Evaluation……71

4.1.2 Train the GNN model……71

5. Summary……72

1. Introduction:

We present in this chapter the implementation of our proposed approach for biological activity prediction, start by describing the data set, advance data processing, increase the data, and present the development environment, detail the different tools used the compilation, 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. Development environment:

2.1 Python:

Python is a high-level, interpreted and general-purpose dynamic programming language that focuses on code readability. The syntax in Python helps the programmers to do coding in fewer steps as compared to Java or C++. The language founded in the year 1991 by the developer Guido Van Rossum has the programming easy and fun to do. The Python is widely used in bigger organizations because of its multiple programming paradigms. They usually involve imperative and object-oriented functional programming. It has a comprehensive and large standard library that has automatic memory management and dynamic features.

2.2 Python Features:

- **Open source:**

Python is publicly available open source software; anyone can use source code that doesn't cost anything.

- **Easy-to-learn:**

Popular (scripting/extension) language, clear and easy syntax, no type declarations, automatic memory management, high-level data types and operations, design to read (more English like syntax) and write (shorter code compared to C, C++, and Java) fast.

- **High-level Language:**

High-level language (closer to human) refers to the higher level of concept from machine language (for example assembly languages). Python is an example of a high-level language like C, C++, Perl, and java with low-level optimization.

- **Portable:**

High level languages are portable, which means they are able to run across all major hardware and software platforms with few or no change in source code. Python is portable and can be used on Linux, Windows, Macintosh, Solaris, FreeBSD, OS/2, Amiga, AROS, AS/400 and many more.

- **Object-Oriented:**

Python is a full-featured object-oriented programming language, with features such as classes, inheritance, objects, and overloading.

- **Python is Interactive:**

Python has an interactive console where you get a Python prompt (command line) and interact with the interpreter directly to write and test your programs. This is useful for mathematical programming.

- **Interpreted:**

A Python program are interpreted, takes source code as input, and then compiles (to portable byte-code) each statement and executes it immediately. No need to compiling or linking.

- **Extendable:**

Python is often referred to as a "glue" language, meaning that it is capable to work in mixed-language environment. The Python interpreter is easily extended and can add a new built-in function or modules written in C/C++/Java code.

- **Libraries:**

Databases, web services, networking, numerical packages, graphical user interfaces, 3D graphics, others.

- **Supports:**

Support from online Python community.

2.3 Python module used:

❖ **Deepchem module :**

The DeepChem project aims to build high quality tools to democratize the use of deep learning in the sciences. The origin of DeepChem focused on applications of deep learning to chemistry, but the project has slowly evolved past its roots to broader applications of deep learning to the sciences.

❖ **Pandas module :**

Pandas is an open source library in Python. It provides ready to use high-performance data structures and data analysis tools. 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.

❖ **Numpy Module:**

NumPy is a Python library used for working with arrays. It also has functions for working in domain of linear algebra, Fourier transform, and matrices .Import the NumPy module:

➢ Import numpy as np .

❖ **Matplotlib Module:**

Matplotlib is a low level graph plotting library in python that serves as a visualization utility and Import the matplotlib module by :

- Import matplotlib.pyplot as plt

❖ **Tensorflow Module:** Deep learning is a machine learning discipline that consists of a set of algorithms inspired by the structure and function of the brain.

TensorFlow is Google's second machine learning framework for designing, building, and training deep learning models. The TensorFlow library may be used to perform numerical computations, which may not appear to be particularly noteworthy in and of itself, but these computations are carried out using data flow graphs. The nodes in these graphs indicate mathematical processes, whereas the edges indicate the data that is transmitted between these links, which are commonly multidimensional data arrays or tensors. The library can run on a variety of CPUs and GPUs, and it is available on a variety of platforms, including mobile.1 We used the newest version of Tensorflow, version 2.5.0, for our practical work in the Google colab environment.

❖ **Keras Module:** Keras is a Python-based high-level neural network API that may be used with TensorFlow, CNTK, or Theano. If you require a deep learning library that allows for easy and quick prototyping, Keras is the way to go (through user friendliness, modularity, and extensibility). Convolutional and recurrent networks, as well as mixtures of the two, are supported. Runs without a hitch on both CPU and GPU ("keras documentation," 2020). In our work, Keras is used to build deep learning models. The version of Keras used is 2.5.0, and Import the Keras module:

- From tensorflow import keras

- **Pytorch Module:** PyTorch is a tensor library intended for usage with GPUs and CPUs in Deep Learning applications. It is a Python-based open-source machine learning package developed mostly by the Facebook AI Research team. It's one of the most popular machine learning libraries, alongside TensorFlow and Keras.
  PyTorch is a tensor computing library based on Python and the Torch library, which performs tensor computations on Graphical Processing Units. Currently, the deep learning and artificial intelligence research community's favorite library.2 As we carried out our practical work on Google colab environment, we use the latest version of Torch which is 1.9.0+cu102.

❖ Python library RDkit

RDkit is a chemoinformatics toolkit with bindings for Python. It is packed with functionality, deployed within multiple open source projects, and is widely used in machine learning applications.

Working with 3D Molecules

The RDkit can generate conformations for molecules using distance geometry, the algorithm followed is:

1. The molecule's distance bounds matrix is calculated based on the connection table and a set of rules.
2. The bounds matrix is smoothed using a triangle-bounds smoothing algorithm.
3. A random distance matrix that satisfies the bounds matrix is generated.
4. This distance matrix is embedded in 3D dimensions (producing coordinates for each atom).
5. The resulting coordinates are cleaned up somewhat using a crude force field and the bounds matrix.

Multiple conformations can be generated by repeating steps 4 and 5 several times, using a different random distance matrix each time.

2.4 Google Colaboratory:

2.4.1 Google Colaboratory definition:

Google Colaboratory widely known as Google Colab is an open source service provided by Google to any person having a Gmail account. Google Colab provides GPU for research to the people who do not have enough resources or cannot afford one. The Google Colab service provides 12.72 GB of RAM and 358.27 GB of hard disk space in one runtime. Every runtime lasts for 12 hours after which the runtime is reset and the user has to establish a connection again. This is to ensure that people do not use the GPU service for crypto currency mining and other illegal purposes. Different run cases are shown below. Once the user opens a Google Colab file they need to select a runtime type. There are 3 available options for the same:

1. None (that will use the computer's CPU the user is using)

2. GPU

3. TPU (especially for tensor processing)

2.4.2 Benefits of Colab:

Besides being easy to use, 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 Tensor Flow, Scikit-learn, Matplotlib among many others are pre-installed and ready to be imported.
- Built on top of Jupiter Notebook.
- Collaboration feature (works with a team just like Google Docs): Google Colab allows Developers to use and share Jupiter 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.

2.5 Google Drive:

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 17: Google Drive applications and functions.

3. Explanation of our approach:

3.1 Dataset:

We used the dataset ones mentioned in the chapter 3, MDDR DS1 database that assign biological activities to compounds.

3.2 Data download :



1	CC[C@H](C)[C@H](NC(=O)C[C@H](O)[C@H](CC1CCCCC1)NC(=O)[C@H](Cc2cccnc2)NC(=O)[C@H](Cc3ccccc3)NC(=O)OC(C)(C)C)C(=O)N[C@@H](CC(C)C)[C@@H](N)CC(=O)OC
2	CC(C)C[C@@H](NC(=O)C\C=C\[C@@H](Cc1ccccc1)NC(=O)OC(C)(C)C)[C@H](O)CC(=O)N[C@H](C)C(=O)N[C@H](CC(C)C)[C@H](O)CC(=O)NCc2ccccc2
3	CC(C)C[C@H](O)[C@H](O)[C@H](CC1CCCCC1)NC(=O)[C@H](Cc2csc(N)n2)NC(=O)[C@H](Cc3ccccc3)NC(=O)C(C)(C)COC(=O)[C@@H](N)CC(=O)O
4	CC(C)C[C@H](O)[C@H](O)[C@H](CC1CCCCC1)NC(=O)[C@H](CC#C)NC(=O)[C@@H](CC(=O)N(C)CCN(C)C)Cc2ccccc2
5	Cl.CC(C)[C@H](NC(=O)C1Cc2ccccc2CN1)C(=O)N[C@@H](CC3CCCCC3)[C@@H](O)CC(=O)N[C@@H](C)C(=O)N[C@@H](Cc4ccccc4)C(=O)N
6	CC(C)C[C@H](O)[C@H](O)[C@H](CC1CCCCC1)NC(=O)[C@H](CC#C)NC(=O)[C@@H](CC(=O)N(C)CCN2CCCCC2)Cc3ccccc3
7	CN(C)CCN(C)C(=O)C[C@@H](Cc1ccccc1)C(=O)N[C@@H](CC#C)C(=O)N[C@@H](CC2CCCCC2)[C@@H](O)[C@@H](O)CC#C
8	CN(C)CCN(C)C(=O)C[C@@H](Cc1ccccc1)C(=O)N[C@@H](Cc2cscn2)C(=O)N[C@@H](CC3CCCCC3)[C@@H](O)[C@@H](O)CC#C
9	CC(C)C[C@H](O)[C@H](O)[C@H](CC1CCCCC1)NC(=O)[C@H](Cc2csc(N)n2)NC(=O)[C@@H]3Cc4ccccc4CN3S(=O)(=O)N5CCOCC5
10	CC(C)C[C@H](O)[C@H](O)[C@H](CC1CCCCC1)NC(=O)[C@H](CC=C)NC(=O)[C@H](Cc2ccccc2)NC(=O)C(C)(C)COC(=O)[C@@H](N)CC(=O)O
11	Cl.CNC1CCN(CC1)C(=O)C[C@@H](Cc2ccccc2)C(=O)N[C@@H](CSC)C(=O)N[C@@H](CC3CCCCC3)[C@@H](O)C(=O)OC(C)C
12	CC(C)C[C@H](O)[C@H](O)[C@H](CC1CCCCC1)NC(=O)C(CC#C)NC(=O)[C@H](Cc2ccccc2)OC(=O)N(C)CCN(C)C
13	CC(C)C[C@H](O)[C@H](O)[C@H](CC1CCCCC1)NC(=O)[C@H](CC#C)NC(=O)[C@@H](CC(=O)N(C)CCN(C)C(C)C)Cc2ccccc2
14	CCOP(=O)(N[C@@H](Cc1cccc2ccccc12)C(=O)N[C@@H](CC(C)C)C(=O)N[C@@H](CC3CCCCC3)[C@@H](O)[C@@H]4CCCO4)OCC
15	CC(C)C[C@H](O)[C@H](O)[C@H](CC1CCCCC1)NC(=O)[C@H](CC#C)NC(=O)[C@H](Cc2ccccc2)CS(=O)(=O)N(C)CCN3CCCCC3
16	CC(C)C[C@H](O)[C@H](O)[C@H](CC1CCCCC1)NC(=O)[C@H](CCC#C)NC(=O)[C@H](Cc2ccccc2)CS(=O)(=O)N(C)CCN3CCCCC3
17	CC(C)C[C@H](O)[C@H](O)[C@H](CC1CCCCC1)NC(=O)[C@H](CC#C)NC(=O)[C@H](Cc2ccccc2)CS(=O)(=O)N(C)CCN3CCC(F)(F)CC3
18	CC(C)C[C@H](O)[C@H](O)[C@H](CC1CCCCC1)NC(=O)[C@H](CCC#C)NC(=O)[C@@H](CC(=O)N(C)CCN2CCCCC2)Cc3ccccc3
19	CC(C)C[C@H](O)[C@H](O)[C@H](CC1CCCCC1)NC(=O)[C@H](CC#C)NC(=O)[C@@H](CC(=O)N(C)CCN2CCC(=O)CC2)Cc3ccccc3
20	CC(C)C[C@H](O)[C@H](O)[C@H](CC1CCCCC1)NC(=O)[C@H](CC#C)NC(=O)[C@@H](CC(=O)N(C)CCN2CCC(F)(F)CC2)Cc3ccccc3

Figure 18: Dataset example from the Dataset.

- The data Contains 8294 compounds divided into eleven groups (Activities) depending on its biological similarities as represented in Table 9, and other datasets for testing the model.

Table 9: our data and it's activities.

N° activity	Start	End	N° compounds
1	1	1130	1130
2	1131	1880	750
3	1881	2683	803
4	2684	3626	943
5	3627	4872	1246
6	4873	5624	752
7	5625	5983	359
8	5984	6378	395
9	6379	7205	827
10	7206	7658	453
11	7659	8294	636

4. Data preprocessing:

4.1 Splitting the data into stratified train and test sets:

The full dataset was randomly divided into two parts, a training set (80% of the dataset, 6635 molecules) to develop the DL model and a test set (20% of the dataset, 1659 molecules) to evaluate the predictive accuracy of the developed model on unseen data.

N° of molecules for training = 6635 molecules.

N° of molecules for testing = 1659 molecules.

Total = 8294 molecules.

```
train_data, test_data = [], []

for _, group_data in papers.groupby("subject"):
    # Select around 50% of the dataset for training.
    random_selection = np.random.rand(len(group_data.index)) <= 0.5
    train_data.append(group_data[random_selection])
    test_data.append(group_data[~random_selection])

train_data = pd.concat(train_data).sample(frac=1)
test_data = pd.concat(test_data).sample(frac=1)

print("Train data shape:", train_data.shape)
print("Test data shape:", test_data.shape)
```

Figure 19: Split dataset to Train / Test.

4.2 Train, validation and test:

Training the model used train dataset. In the case of neural networks, the model learns its weights and biases.

The model uses validation dataset for evaluation after every set of predictions as well as helps the model tune its hyper-parameters.

The test dataset is used to evaluate the model after it has been completely trained.

4.3 Network Architecture :

4.3.1 GNN implementation:

Implement Train and Evaluate Experiment.

```
hidden_units = [32, 32]
learning_rate = 0.01
dropout_rate = 0.5
num_epochs = 300
batch_size = 256
```

Figure 20: Implement Train and Evaluate Experiment.

This function compiles and trains an input model using the given training data.

```
def run_experiment(model, x_train, y_train):
    # Compile the model.
    model.compile(
        optimizer=keras.optimizers.Adam(learning_rate),
        loss=keras.losses.SparseCategoricalCrossentropy(from_logits=True),
        metrics=[keras.metrics.SparseCategoricalAccuracy(name="acc")],
    )
    # Create an early stopping callback.
    early_stopping = keras.callbacks.EarlyStopping(
        monitor="val_acc", patience=50, restore_best_weights=True
    )
    # Fit the model.
    history = model.fit(
        x=x_train,
        y=y_train,
        epochs=num_epochs,
        batch_size=batch_size,
        validation_split=0.15,
        callbacks=[early_stopping],
    )

    return history
```

Figure 21: compiles and trains an input model.

This function displays the loss and accuracy curves of the model during training.

```
def display_learning_curves(history):
    fig, (ax1, ax2) = plt.subplots(1, 2, figsize=(15, 5))

    ax1.plot(history.history["loss"])
    ax1.plot(history.history["val_loss"])
    ax1.legend(["train", "test"], loc="upper right")
    ax1.set_xlabel("Epochs")
    ax1.set_ylabel("Loss")

    ax2.plot(history.history["acc"])
    ax2.plot(history.history["val_acc"])
    ax2.legend(["train", "test"], loc="upper right")
    ax2.set_xlabel("Epochs")
    ax2.set_ylabel("Accuracy")
    plt.show()
```

Figure 22: the model during of training.

4.3.2 Train the GNN model:

Note that we use the standard *supervised* cross-entropy loss to train the model. However, we can add another *self-supervised* loss term for the generated node embeddings that makes sure that neighbouring nodes in graph have similar representations, while faraway nodes have dissimilar representations.

```
x_train = train_data.paper_id.to_numpy()
history = run_experiment(gnn_model, x_train, y_train)
```

Figure 23: Train the GNN model.

Using 300 Epochs :

```
Epoch 1/300
5/5 [==============================] - 4s 188ms/step - loss: 2.2529 - acc: 0.1793 - val_loss:
Epoch 2/300
5/5 [==============================] - 0s 83ms/step - loss: 1.9866 - acc: 0.2601 - val_loss: 1
Epoch 3/300
5/5 [==============================] - 0s 77ms/step - loss: 1.8794 - acc: 0.2846 - val_loss: 1
Epoch 4/300
5/5 [==============================] - 0s 74ms/step - loss: 1.8432 - acc: 0.3078 - val_loss: 1
Epoch 5/300
5/5 [==============================] - 0s 69ms/step - loss: 1.8314 - acc: 0.3134 - val_loss: 1
Epoch 6/300
5/5 [==============================] - 0s 68ms/step - loss: 1.8157 - acc: 0.3208 - val_loss: 1
Epoch 7/300
5/5 [==============================] - 0s 94ms/step - loss: 1.8112 - acc: 0.3071 - val_loss: 1
Epoch 8/300
5/5 [==============================] - 0s 67ms/step - loss: 1.8028 - acc: 0.3132 - val_loss: 1
Epoch 9/300
5/5 [==============================] - 0s 68ms/step - loss: 1.8007 - acc: 0.3206 - val_loss: 1
Epoch 10/300
5/5 [==============================] - 0s 68ms/step - loss: 1.7571 - acc: 0.3259 - val_loss: 1
Epoch 11/300
5/5 [==============================] - 0s 68ms/step - loss: 1.7373 - acc: 0.3279 - val_loss: 1
Epoch 12/300
5/5 [==============================] - 0s 76ms/step - loss: 1.7130 - acc: 0.3169 - val_loss: 1
```

Figure 24: Epochs of GNN model.

Let's plot the learning curves.

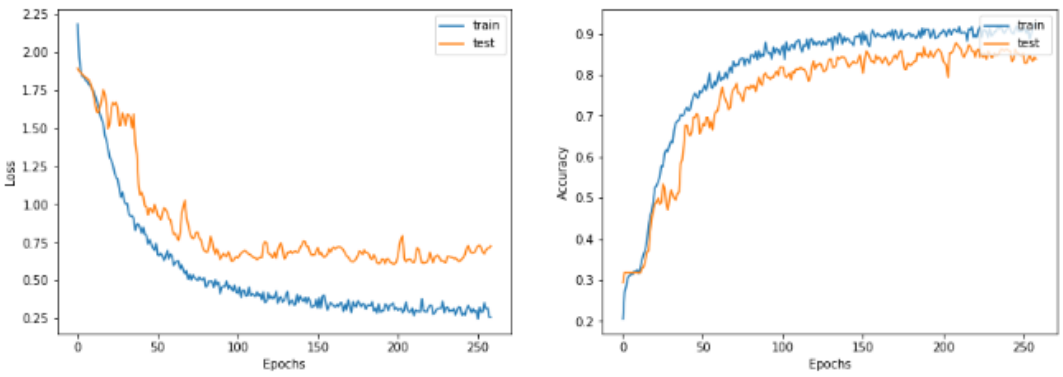


Figure 25: The training and validation loss..

Now we evaluate the baseline model on the test data split.

```
_, test_accuracy = baseline_model.evaluate(x=x_test, y=y_test, verbose=0)
print(f"Test accuracy: {round(test_accuracy * 100, 2)}%")
```

Figure 26: evaluate the baseline model on the test data split.

```
Test accuracy: 73.52%
```

Figure 27: The result of the evaluation.

4.4 Evaluation:

4.4.1 GNN model:

After conducting experiments in model GNN. We were able to get the results shown in the Figure 4 plots the predicted of samples in the training and test sets against the corresponding experimental epochs

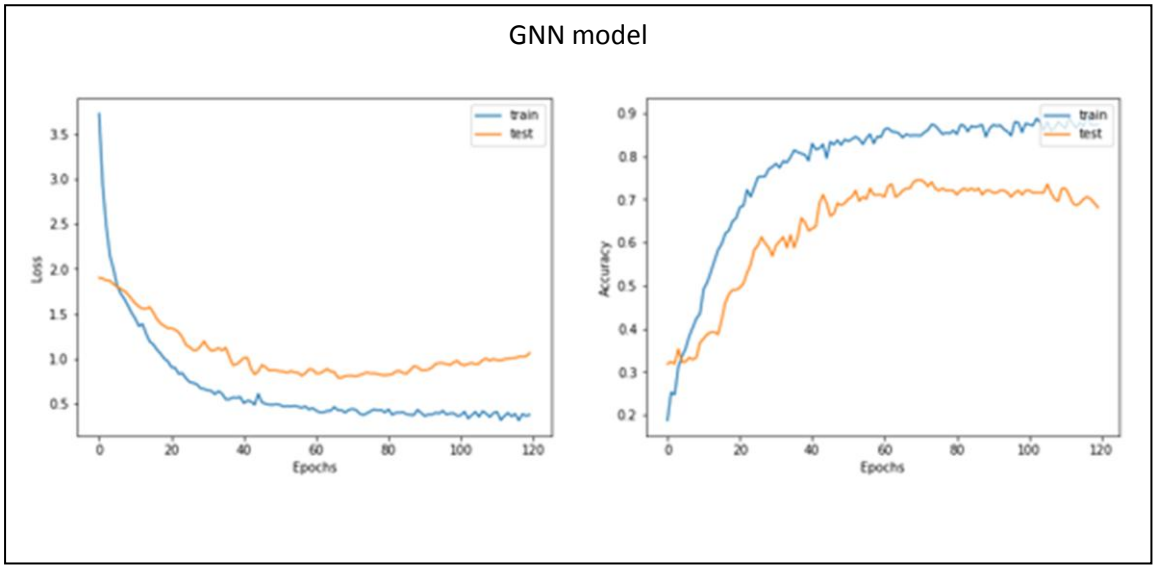


Figure 28: Comparison between train and test for accuracy and for loss in GNN.

5. Summary:

In this chapter, we presented the implementation, the tests and the obtained results of our proposed approach, First of all, it is divided into two parts:

The first part is about data such as pre-processing, splitting the data into groups, while the second part relates to the model as training, compiling and testing the model and getting the results. Which mainly aims to improve the process of biological activity prediction, based on deep learning Graph neural network (GNN).

General conclusion:

Over the past few years, graph neural networks have become powerful and practical tools for any problem that can be modeled by graphs.

Quantitative activity prediction models were constructed the application of deep learning methods to the prediction of the biological activity and physical properties of molecules in drug discovery programs. Rather than employing the expert-derived chemical features typically used to build predictive models, researchers are now using deep learning to develop novel molecular representations. These representations, coupled with the ability of deep neural networks to uncover complex, nonlinear relationships, have led to state-of-the-art performance. While deep learning has changed the way that many researchers approach QSARs, it is not a panacea. As with any other machine learning task, the design of predictive models is dependent on the quality, quantity, and relevance of available data. Seemingly fundamental issues, such as optimal methods for creating a training set, are still open questions for the field. Ensemble learning improved the predictive performance compared to the individual models.

Even if the targets on which activity prediction models are built are "unappealing", the models can provide useful hints for drug repositioning, alerting to potential off-targets, prioritizing strategies in the early stage of drug development, finding poly-pharmacological drugs, and searching for tool compounds that support the elucidation of the molecular mechanisms underlying biological function. From this point of view, we believe that a model that ranks compounds not by binary classification but by quantitative prediction is a useful tool in drug discovery research. We believe that our GNN architecture could play a crucial part in such an effort,The GNN model used are trained, validated, and tested based on a dataset of MDDR DS1.

References:

[1]: Nantasenamat C, Isarankura-Na-Ayudhya C, Naenna T, Prachayasittikul V. A practical overview of quantitative structure-activity relationship. EXCLI J. 2009;8:74–88.

[2]: Hasana AAM. Similarity-based virtual screening using Bayesian inference network for searching chemical database. 2010;(March):191.

[3]: Walters WP, Barzilay R. Applications of Deep Learning in Molecule Generation and Molecular Property Prediction. Acc Chem Res. 2021 Jan 19;54(2):263-270. doi:10.1021/acs.accounts.0c00699. Epub 2020 Dec 28. PMID: 33370107.

[4]: Mahobia N, Patel R, Sheikh D, Singh S, Mishra A, Dhardubey R. Validation method used in quantitative structure activity relationship. Der Pharma Chem. 2010 Jan 1;2:260–71.

[5]: Patel RD, Singh S, Mishra A. Scholars Research Library. 2010;(January).

[6]: Willett P. Similarity Searching Using 2D Structural Fingerprints. Methods Mol Biol. 2011 Jan 1;672:133–58.

[7]: Ali Y, Ali A, Ali Y, Ali A. FOR SEARCHING CHEMICAL DATABASE UNIVERSITI TEKNOLOGI MALAYSIA DECLARATION OF THESIS / UNDERGRADUATE PROJECT PAPER AND COPYRIGHT.

[8]: Halder AK, Moura AS, Cordeiro MNDS. QSAR modelling: a therapeutic patent review 2010-present. Expert Opin Ther Pat. 2018 Jun;28(6):467-476. doi: 10.1080/13543776.2018.1475560. Epub 2018 May 23. PMID: 29745270.

[9]: Xu Y. Deep Neural Networks for QSAR. Methods Mol Biol. 2022;2390:233-260. doi: 10.1007/978-1-0716-1787-8_10. PMID: 34731472.

[10]: Brusco M, Cradit D, Steinley D. A comparison of 71 binary similarity coefficients: The effect of base rates. PLoS One. 2021 Apr 7;16:e0247751.

[11]: Kwon S, Bae H, Jo J, Yoon S. Comprehensive ensemble in QSAR prediction for drug discovery. 2019;1–12.

[12]: Berhanu W, Pillai GG, Oliferenko A, Katritzky A. Quantitative Structure–Activity/Property Relationships: The Ubiquitous Links between Cause and Effect. Chempluschem. 2012 Jul 1;77:507–17.

[13]: Kwon S, Bae H, Jo J, Yoon S. Comprehensive ensemble in QSAR prediction for drug discovery. 2019;1–12.

[14]: Pradhyumna P, Shreya GP. Graph Neural Network (GNN) in Image and Video Understanding Using Deep Learning for Computer Vision Applications. 2021;1183–9.

[15]: Ma J, Sheridan R, Liaw A, Dahl G, Svetnik V. Deep Neural Nets as a Method for Quantitative Structure–Activity Relationships. J Chem Inf Model. 2015 Jan 30;55.

[16]: Halder AK, Moura AS, Cordeiro MNDS. QSAR modelling: a therapeutic patent review 2010-present. Expert Opin Ther Pat. 2018 Jun;28(6):467-476. doi: 10.1080/13543776.2018.1475560. Epub 2018 May 23. PMID: 29745270.

[17]: Alvarsson J, Wikberg JES, Spjuth O, Andersson C, Eklund M. Ligand-based Methods for Data Management and Modelling [Internet]. Digital Comprehensive Summaries of Uppsala Dissertations from the Faculty of Pharmacy. 2015. 73 p. Availablefrom:http://urn.kb.se/resolve?urn=urn:nbn:se:uu:diva248964%0Ahttp://uu.divaportal.org/smash/get/diva2:801538/FULLTEXT02.pdf%0Ahttp://uu.diva-portal.org/smash/get/diva2:801538/PREVIEW01.jpg

[18]: Perkins, R., Fang, † Hong, Tong, W., & Welsh, W. J. (2003). Annual Review QUANTITATIVE STRUCTURE-ACTIVITY RELATIONSHIP METHODS: PERSPECTIVES ON DRUG DISCOVERY AND TOXICOLOGY. In *Environmental Toxicology and Chemistry* (Vol. 22, Issue 8).

[19]: Patel RD, Singh S, Mishra A. Scholars Research Library. 2010;(January).

[20]: Zhou J, Cui G, Hu S, Zhang Z, Yang C, Liu Z, et al. Graph neural networks: A review of methods and applications. AI Open. 2020 Jan 1;1:57–81.

[21] : A. Abdo and N. Salim. Bayesian Inference Network for Molecular Similarity Searching Using 2D Fingerprints and Multiple-activity Molecules. Jurnal Teknologi Maklumat, 2008. 20(3):1-13.

[22] : A. Abdo and N. Salim, Inference Networks for Chemical Similarity Searching. International Conference on Advanced Computer Theory and Engineering, 2008, 408-412.

[23]: Holliday, J. D., Salim, N., and Wilett, P., On the magnitudes of coefficient values in the calculation of chemical similarity and dissimilarity, In: Laine, B. K., Editor. Chemometrics and Chemoinformatics. Washington, D.C. ACS Chemical Society.77-95; 2005.

[24]: Brown, R. D. and Martin, Y. C., The Information Content of 2D and 3D Structural Descriptors Relevant to Ligand-Receptor Binding. Journal of Chemical Information and Computer Sciences, 1997. 37(1): 1-9.

[25]: Chen, X. and Reynolds, C. H., Performance of Similarity Measures in 2D Fragment-Based Similarity Searching: Comparison of Structural Descriptors and Similarity Coefficients. Journal of Chemical Information and Computer Sciences, 2002. 42(6): 1407-1414.

[26]: Ormerod, A., Willett, P., and Bawden, D., Comparison of Fragment Weighting Schemes for Substructural Analysis. Quantitative StructureActivity Relationships, 1989. 8(2): 115-129 [175]: Ormerod, A., Willet, P., and Bawden, D., Further Comparative Studies of Fragment Weighting Schemes for Substructural Analysis. Quantitative Structure-Activity Relationships, 1990. 9(4): 302-312.