

**VIET NAM NATIONAL UNIVERSITY HO CHI MINH CITY  
HO CHI MINH CITY UNIVERSITY OF TECHNOLOGY  
FACULTY OF COMPUTER SCIENCE AND ENGINEERING**



**GRADUATION THESIS**

**AI-POWERED DECISION SUPPORT SYSTEM FOR  
ANTIVIRAL PHARMACEUTICAL FORMULATION**

Major: COMPUTER SCIENCE

<b>COUNCIL</b>	<b>: COMPUTER SCIENCE 6</b>
<b>SUPERVISORS</b>	<b>: ASSOC. PROF. QUAN THANH THO MR. BANG NGOC BAO TAM</b>
<b>REVIEWER</b>	<b>: MR. VO THANH HUNG</b>
<b>STUDENT</b>	<b>: NGUYEN QUANG DUC - 1810118</b>

HO CHI MINH CITY, MAY 2022

KHOA: KH & KT Máy tính  
BỘ MÔN: KHMT

**NHIỆM VỤ LUẬN ÁN TỐT NGHIỆP**  
*Chú ý: Sinh viên phải dán tờ này vào trang nhất của bản thuyết trình*

HỌ VÀ TÊN: NGUYỄN QUANG ĐỨC  
NGÀNH: KHOA HỌC MÁY TÍNH

MSSV: 1810118  
LỚP: MT18KHTN

**1. Đầu đề luận án:**

Hệ thống trí tuệ nhân tạo hỗ trợ điều chế thuốc kháng vi-rút

**2. Nhiệm vụ (yêu cầu về nội dung và số liệu ban đầu):**

- Thu thập và xử lý dữ liệu thử sai của thuốc đối với vi-rút (vi-rút SAR-COV-2)
- Nghiên cứu xây dựng mô hình dự đoán, hỗ trợ thiết kế thuốc
- Đánh giá tính hiệu quả của mô hình trong điều kiện thực tế
- Xây dựng ứng dụng với giao diện đề tương tác với mô hình

**3. Ngày giao nhiệm vụ luận án: 10/01/2022**

**4. Ngày hoàn thành nhiệm vụ: 03/06/2022**

**5. Họ tên giảng viên hướng dẫn:**

**Phản hướng dẫn:**

1) Quân Thành Thơ

2) Mai Đức Trung *Bảng Ngọc bảo Tâm*

3)

Nội dung và yêu cầu LVTN đã được thông qua Bộ môn.

Ngày ..... tháng ..... năm .....

**CHỦ NHIỆM BỘ MÔN**  
(Ký và ghi rõ họ tên)

**GIẢNG VIÊN HƯỚNG DẪN CHÍNH**  
(Ký và ghi rõ họ tên)



**PGS.TS. Quân Thành Thơ**

*PHẦN DÀNH CHO KHOA, BỘ MÔN:*

Người duyệt (chấm sơ bộ):

Đơn vị:

Ngày bảo vệ:

Điểm tổng kết:

Nơi lưu trữ luận án:

-----  
Ngày      tháng      năm

## PHIẾU CHẤM BẢO VỆ LVTN

(Dành cho người hướng dẫn/phản biện)

1. Họ và tên SV: Nguyễn Quang Đức  
MSSV: 1810118      Ngành (chuyên ngành): KHMT
2. Đề tài: Hệ thống trí tuệ nhân tạo hỗ trợ điều chế thuốc kháng vi-rút
3. Họ tên người hướng dẫn/phản biện: Quản Thành Thơ
4. Tổng quát về bản thuyết minh:  
Số trang: 107      Số chương: 6  
Số bảng số liệu: 15      Số hình vẽ: 38  
Số tài liệu tham khảo: 77      Phần mềm tính toán:  
Hiện vật (sản phẩm):
5. Tổng quát về các bản vẽ:  
- Số bản vẽ:      Bản A1:      Bản A2:      Khổ khác:  
- Số bản vẽ vẽ tay      Số bản vẽ trên máy tính:
6. Những ưu điểm chính của LVTN:
  - Luận văn nghiên cứu một bài toán có tính học thuật vào, kết hợp tri thức của cả hai mảng Khoa học Máy tính và Hóa Sinh
  - Luận văn đã đề xuất ra một mô hình học máy mới kết hợp một trong những mô hình và kỹ thuật học sâu tiên tiến nhất với các kiến thức chuyên dụng về hóa sinh.
  - Luận văn cũng đã có một hệ thống giao diện trực quan thuận tiện cho người dùng sử dụng.
  - Công việc của luận văn đã được viết thành một bài báo khoa học gửi phản biện ở một trong những hội nghị lớn nhất về thiết kế thuốc.
7. Những thiếu sót chính của LVTN:

8. Đề nghị: Được bảo vệ       Bổ sung thêm để bảo vệ       Không được bảo vệ

9. 3 câu hỏi SV phải trả lời trước Hội đồng:

a.

b.

c.

10. Đánh giá chung (bằng chữ: giỏi, khá, TB):

Điểm: 9.5/10

Ký tên (ghi rõ họ tên)



PGS.TS. Quản Thành Thơ



-----  
Ngày 30 tháng 5 năm 2022

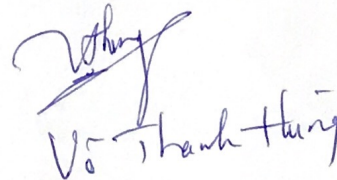
## PHIẾU CHẤM BẢO VỆ LVTN

(Dành cho người hướng dẫn/phản biện)

1. Họ và tên SV: Nguyễn Quang Đức      MSSV: 1810118
- Ngành (chuyên ngành): Khoa học máy tính
2. Đề tài: **AI-Powered Decision Support System for Antiviral Pharmaceutical Formulation**
3. Họ tên người hướng dẫn/phản biện: ThS. Võ Thanh Hùng
4. Tổng quát về bản thuyết minh:
- |                           |                     |
|---------------------------|---------------------|
| Số trang: 97              | Số chương: 6        |
| Số bảng số liệu: 15       | Số hình vẽ: 37      |
| Số tài liệu tham khảo: 77 | Phần mềm tính toán: |
- Hiện vật (sản phẩm) ứng dụng web
5. Tổng quát về các bản vẽ:
- |                           |                          |           |
|---------------------------|--------------------------|-----------|
| - Số bản vẽ:      Bản A1: | Bản A2:                  | Khổ khác: |
| - Số bản vẽ vẽ tay        | Số bản vẽ trên máy tính: |           |
6. Những ưu điểm chính của LVTN:
- Luận văn xây dựng một hệ thống hỗ trợ ra quyết định nhằm xây dựng công thức dược phẩm kháng virus, tức hỗ trợ điều chế thuốc điều trị một bệnh nào đó (ví dụ Covid-19). Đây là một đề tài có ý nghĩa thực tiễn cao.
- Sinh viên đã đề xuất 2 bước xử lý bao gồm xác định sự tương tác giữa Ligands và Protein, và sử dụng thông tin này để xây dựng mô hình cavity dùng cho hệ thống tiến cử (recommendation) bằng rule-based. Sự đề xuất của tác giả là phù hợp khi việc thu thập dữ liệu thực tế cho lĩnh vực này là cực kỳ khó khăn. Tuy chưa giải quyết triệt để, và việc tích lũy lỗi sẽ diễn ra khi thực hiện theo mô hình này, đề tài là một bước khởi đầu cho những cải tiến trong tương lai.
- Kết quả luận văn đã được viết thành một bài báo khoa học chuyên ngành, đây là kết quả đáng khích lệ cho một luận văn đại học.
7. Những thiếu sót chính của LVTN:
8. Đề nghị: Được bảo vệ       Bổ sung thêm để bảo vệ       Không được bảo vệ
9. Câu hỏi SV phải trả lời trước Hội đồng:
- a.
10. Đánh giá chung (bằng chữ: giỏi, khá, TB):

Điểm :    9/10

Ký tên (ghi rõ họ tên)

  
Võ Thanh Hùng



# Declaration of Authenticity

I declare that this research is my own work, conducted under the supervision and guidance of Assoc. Prof. Dr. Quan Thanh Tho and Mr. Bang Ngoc Bao Tam. The result of our research is legitimate and has not been published in any forms prior to this. All materials used within this researched are collected myself by various sources and are appropriately listed in the references section.

In addition, within this research, I also used the results of several other authors and organizations. They have all been aptly referenced.

In any case of plagiarism, I stand by my actions and will be responsible for it. University of Technology - Vietnam National University HCMC therefore are not responsible for any copyright infringements conducted within our research.

Ho Chi Minh City, December 2021

Author

Nguyen Quang Duc

# Acknowledgements

Firstly, I would like to express my deep and sincere thanks to my supervisor, Dr. Quan Thanh Tho, for his patience, guidance, and support in almost my student years. I have benefited greatly from your wealth of knowledge and meticulous editing. I am extremely grateful that you took me on as a student and continued to have faith in me over the years.

Secondly, I gratefully recognize the help of Prof. Truong Nguyen Thanh - The University of Utah, Dr. Nguyen Thuy Viet Phuong - University of Medicine and Pharmacy at HCMC. Without their support, I could not complete this meaningful research.

Nextly, I am very thankful for my predecessors especially Mr. Nguyen Hoang Tuan and Mr. Duong Quoc Cuong. Their advice and encouragements are precious things I need to preserve. I also give my special thank to my successors including Mr. Nguyen Ho Quang, Mr. Ly Thanh Bach, Mr. Le Duc Khoan, and Mr. Nguyen Dinh An for supporting me in research and implementation.

My family deserves endless gratitude for their everlasting love, sacrifice, and support keeping me motivated and confident. My accomplishments and success are because they believed in me. There is no word that can describe my love for my family.



# Abstract

In the context of the Coronavirus (COVID-19) pandemic that is resurgent globally with dangerous Delta, and Omicron variants, the number of cases and deaths is increasing at a dizzying rate. As a consequence, doctors and nurses are having to work day and night against the effects of this pandemic. To lower the mortality rate and lower pressure on the medical staff, the most anticipated thing right now is vaccines and drugs. The story of vaccine development is probably not a new one, and we can rely on previous techniques to synthesize and prepare vaccines. Thus, many vaccines have produced with high protection ability such as Pfizer, AstraZeneca, Sputnik V, etc. However, it is a completely different story for drugs, because the structural differences of different virus strains make it difficult to find viral inhibitory proteins. This is basically based on the interactions between the atoms of virus main proteins and the atoms present in the drug. The obstacle here is there are only one or a few drugs that can inhibit. This will lead to scientists having to try and fail many times to find the right formula for making drugs. This process somehow can take up to 10 years with billions of USD required. In the race against virus variant proliferation, drug development needs to be sped up. Therefore, an Artificial Intelligence (AI) system with knowledge gained from prior trial and error attempts can be used to predict necessary atoms and their structure in the drug to achieve better modulation results. In this thesis, we create an AI-powered decision support system for assisting drug designing process. With our system, the average binding affinity of designed drugs has been improved by  $-0,762 \text{ kcal/mol}$ . As a result, this system also helps increase the speed of researchers, shortening the time of in vitro screening to quickly get drugs to patients who are waiting for them day by day, hour by hour.

# Contents

<b>Declaration of Authenticity</b>	<b>v</b>
<b>Acknowledgements</b>	<b>vi</b>
<b>Abstract</b>	<b>vii</b>
<b>1 Introduction</b>	<b>1</b>
1.1 Problem Statement . . . . .	1
1.2 Goals . . . . .	2
1.3 Scope . . . . .	3
1.4 Thesis Structure . . . . .	3
<b>2 Theoretical Background</b>	<b>4</b>
2.1 Pharmaceutical Knowledge . . . . .	4
2.2 Decision Support System in Pharmaceutical Formulation . . . . .	8
2.3 Target Cavity Model for Drug-designing Suggestion . . . . .	10
2.4 Drug-Target Interaction Problem . . . . .	12
2.5 Molecular Representation . . . . .	13
2.6 Graph Attention Network . . . . .	15
2.7 Transfer Learning Strategy . . . . .	18
2.8 Nearest Neighbors . . . . .	19
2.9 DBSCAN Algorithm . . . . .	22
<b>3 Related Works</b>	<b>23</b>
3.1 <i>De novo</i> drug design . . . . .	23



3.2	Computer-Aided Drug Design Systems . . . . .	26
3.3	Docking Tools . . . . .	29
3.4	Drug-Target Interaction Prediction . . . . .	30
<b>4</b>	<b>The Proposed Solutions</b>	<b>32</b>
4.1	Overview . . . . .	32
4.2	Detail of components . . . . .	35
4.2.1	Identifying Drug-Protein Interactions . . . . .	35
4.2.2	Building Cavity Model for Target Protein . . . . .	49
4.2.3	Building Algorithm for Recommending . . . . .	51
4.3	Web application . . . . .	53
4.3.1	Overview of Application . . . . .	53
4.3.2	Requirements . . . . .	55
4.3.3	Usecase Descriptions . . . . .	55
4.3.4	Database Design . . . . .	60
<b>5</b>	<b>Experiments and Results</b>	<b>62</b>
5.1	Performance of The Propsoed GNN-based Model . . . . .	62
5.1.1	Pharmacologically Active Predicting . . . . .	62
5.1.2	Drug-Protein Pairwise Interactions Prediction . . . . .	67
5.2	The Cavity Model of Target Protein . . . . .	68
5.3	The Performance of Decision Support System . . . . .	70
5.4	Web Application . . . . .	72
5.4.1	Screenshots on Multiple Platforms . . . . .	72
5.4.2	Computing Time . . . . .	77
<b>6</b>	<b>Conclusion</b>	<b>79</b>
6.1	Summary . . . . .	79
6.2	Future Developments . . . . .	80
	<b>References</b>	<b>81</b>
	<b>A Scientific paper</b>	<b>90</b>

# List of Figures

1.1	The current situation of ViDok system . . . . .	2
2.1	An example illustrating the concept of "strong" and "weak" ligand .	6
2.2	Overall process of drug design . . . . .	6
2.3	The overview of a typical decision support system . . . . .	9
2.4	An example of pharmacophore model containing different features of a protein [1] . . . . .	11
2.5	An example of <i>Hydrogen</i> and <i>Hydrophobic</i> interactions between a lig- and and the target protein . . . . .	12
2.6	Some examples of common molecular representations . . . . .	14
2.7	Conceptualization of a GNN layer . . . . .	15
2.8	An overview of transfer learning strategy . . . . .	18
2.9	An example of using Nearest Neighbors algorithm. The available points are colored as red while the new point is colored as green and the cut-off distance equals 2. . . . .	19
2.10	An example of building KDTree from a dataset with different stop conditions . . . . .	21
2.11	An example of DBSCAN clustering two regions . . . . .	22
3.1	A schematic graph representing current <i>de novo</i> drug design approaches	24
3.2	The screenshot of the COVID Moonshot system . . . . .	27
3.3	The screenshots of the Vidok system . . . . .	28
3.4	An example of docking process . . . . .	29
4.1	The overview of our system following the typical DSS . . . . .	33



4.2	The overview of our system together with the ViDok system . . . . .	34
4.3	The overall pipeline of Identifying Drug-Protein Interactions component	35
4.4	The architecture of the baseline model . . . . .	39
4.5	The architecture of our modified model . . . . .	42
4.6	Illustration of the normal flow and transfer learning flow used in this study . . . . .	43
4.7	The difference between models with and without multi-hop gating mechanism . . . . .	45
4.8	An example using our proposed Nearest Neighbors based algorithm .	48
4.9	An example of pharmacophore model flattened in 2D space . . . . .	49
4.10	Our drug design system integrated AI-powered decision support system	54
4.11	Overview of all usecases in our application . . . . .	56
4.12	EERD of database . . . . .	60
4.13	Database design schema . . . . .	61
5.1	Our built pharmacophore model . . . . .	68
5.2	Our predicted interacting protein atoms using top 1000 complexes from ViDok. The blue, red and gray colors indicate <i>Hydrogen Donor</i> , <i>Hydrogen Acceptor</i> and <i>Hydrophobic</i> interaction, respectively. . . . .	69
5.3	Histogram of docking scores between ligands with and without suggestion	71
5.4	Screenshot of the ligand designing tool . . . . .	72
5.5	Screenshot of the suggestion area . . . . .	73
5.6	Screenshot of the result area . . . . .	74
5.7	Screenshot of the scoreboard . . . . .	75
5.8	Screenshots of our web application on multiple platforms . . . . .	76
5.9	Histogram of processing time for ligands with and without suggestion	78

# List of Tables

4.1	The list of atom features used in original study and in Improvement 1	38
4.2	The list suitable atoms for recommending by each chemical rules . . .	51
4.3	”Log in” usecase description . . . . .	57
4.4	”Log out” usecase description . . . . .	57
4.5	”Register” usecase description . . . . .	58
4.6	”Design ligand” usecase description . . . . .	58
4.7	”Apply suggestion” usecase description . . . . .	59
4.8	”View designed ligands” usecase description . . . . .	59
5.1	The number of protein-compound complexes used for training and testing of each dataset . . . . .	63
5.2	AUC scores of baseline model compared to other models in various settings grouped by molecular representation type . . . . .	64
5.3	The evaluation results of pairwise interaction prediction . . . . .	67
5.4	Built pharmacophore model of the target protein . . . . .	69
5.5	Information about participants in our experiment . . . . .	70
5.6	Experiment results about the performance of our decision support system	71
5.7	The results of our time-tesing experiment . . . . .	77

# Chapter 1

## Introduction

### 1.1 Problem Statement

The *Coronavirus* (COVID-19) is currently at a rapid development speed. Many of its variants have been considered dangerous by *World Health Organization* (WHO) such as *Beta*, *Delta*, and recently *Omicron*. In contrast with the virus transformation speed, the race for developing drugs is in a sustainable situation. Unlike the process of vaccine development, experts can follow many previous techniques to make a candidate vaccine, in drug development, a candidate drug is considered not only pharmacologically active with the receptor the virus attaches to but also safe for humans [2]. Among all the drugs humans can make, the number of drugs that satisfy all conditions can be counted on the head of fingers [3]. Therefore, an intelligent assistant is an idea to speed up the process of drug design.

The ViDok system [4] is an open, community-based drug-designing system developed by a research group of the University of Utah. This system contains a mobile application for end users to design candidate drugs and a backend for processing designed drugs from users. Using the mobile application, anyone can create his own drugs and submit them to the ViDok system. The ViDok system, then, perform docking and binding affinity calculation to rank the drugs user submitted. The obstacle in designing drugs is that not many users have enough chemical knowledge to make good drugs (drugs that have high ranking). To take advantage of community power, we need an intelligent assistant that can give suggestions to the users with

less chemical knowledge to help them make better drugs. Figure 1.1 below illustrates the current situation of drug design on the ViDok system.

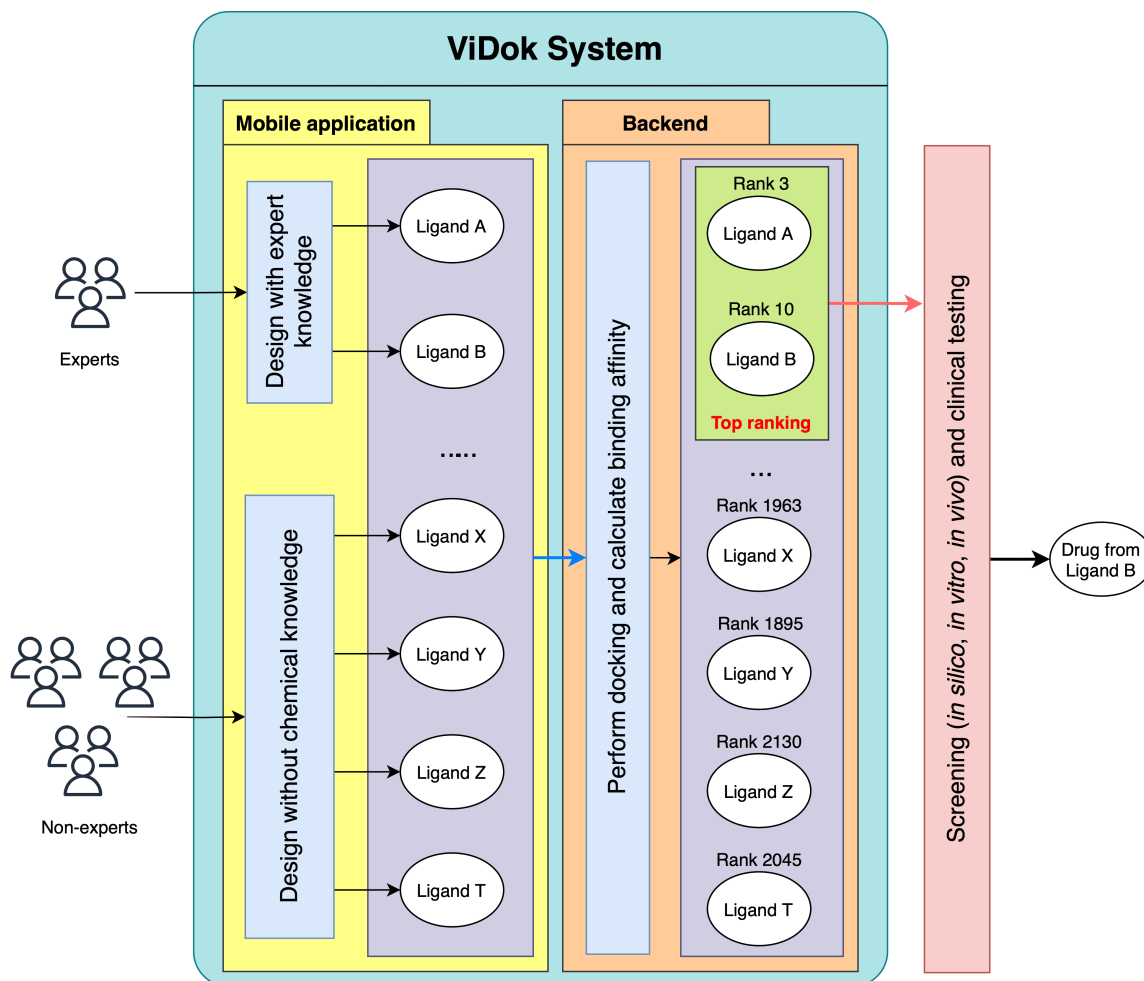


Figure 1.1: The current situation of ViDok system

## 1.2 Goals

This thesis is about building a decision support system for the ViDok system to help users create better drugs. Our system will take input of user-designed drugs, then analyze their interactions with the target receptor and give back users drug-designing

suggestions to make their designed drugs better. Multiple Artificial Intelligence techniques are expected to be used in our system to make predictions of necessary information for creating drug-building recommendations.

### **1.3 Scope**

In this thesis, we will design and build an efficient high accurate drug-building decision support system. Besides that, we also study and implement Artificial Intelligence techniques and other algorithms required for our system. The final result will be a running decision support system ready for being integrated into the ViDok system.

### **1.4 Thesis Structure**

There are totally five chapters in this thesis. The first chapter is a brief view of the problem we are solving and our vision. The second chapter makes clearer about chemical knowledge used in this thesis and gives insights on some related Artificial Intelligence techniques. Then the related works about drug design and computer-aided drug design system are presented in Chapter 3. The next chapter is the soul of this thesis. In that chapter, we describe our methods to create a drug design suggestions step by step with high detail and our web-based system. Chapter Experiments presents our validation results of de The final chapter is a brief summarization of this thesis and discussions on future developments of the designed system.

# Chapter 2

## Theoretical Background

### 2.1 Pharmaceutical Knowledge

This section provides detail of necessary chemical knowledge used in this thesis. It consists of some basic definitions, basic concepts of designing drugs and atoms interactions.

Firstly, the definition of virus, target, ligand and drug are in following together with a theorem about the interaction between target and drug (Definition 1, 2, 3, 4 and Theorem 1).

**Definition 1** *Virus* is the causative agents of infectious diseases. It works by entering the organism's cell and reproducing.

**Definition 2** *Target* is one of the virus's main specific proteins, which are mediation for the assembly of replication-transcription machinery, in order to form new kernels for child viruses. To stop the virus activity, it is ideal to attack the target.

**Definition 3** *Ligand* is a set of atoms that are linked together in some way and can cause some effects on specific proteins. Ligand is sometime called compound.

**Definition 4** *Drug* is a medicine that can cause some chemical effects on the target. Drug is formed by a main ligand with appropriate adjuvants.

**Theorem 1** A target can only be pharmacologically active with one or a few specific ligands.



When a ligand interacts with the target protein, it creates almost interactions in the cavity of the protein (Definition 5). This assumption has been proved by Johnson and Karanicolas [5]. The set of atoms in the protein that usually occurs interaction is called binding site (Definition 6). Due to the specified properties of the protein cavity, we then can infer that there are a very small number of ligands that have affects on the target virus's replication process (Theorem 1). Among all those drugs, there are only a few drugs that have enough "strength" to stop the virus from reproducing. Based on that, a drug design process is defined in Definition 7. Figure 2.1 shows an example of "strong" and "weak" ligand. In case of "strong" ligand, it can make the target protein inactivated, thus, the virus will stop reproducing.

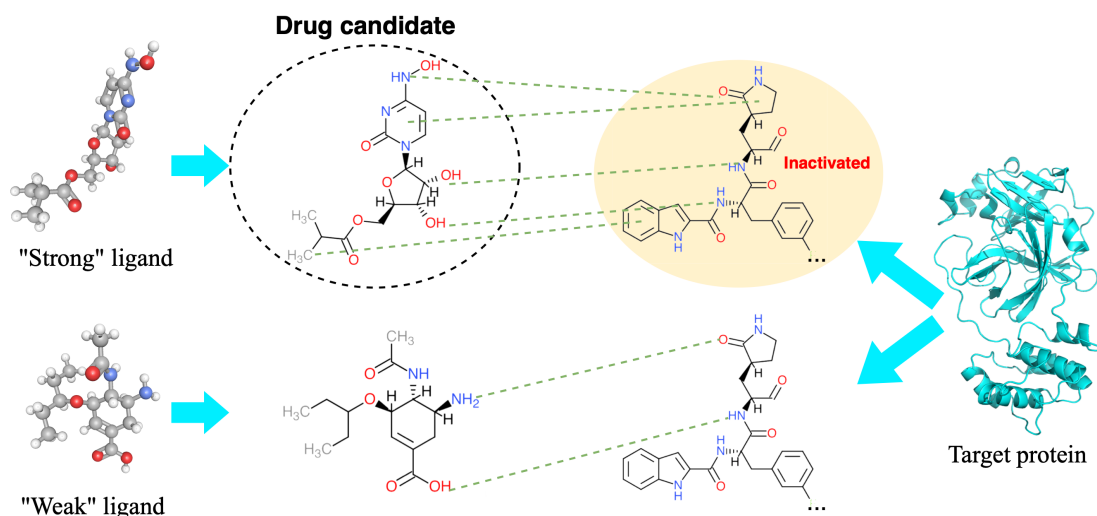
**Definition 5** *Cavity*, which is also known as pocket, is a set of atoms in an area of a protein. These atoms are considered characteristic for that protein, which means there are usually interactions that occur to these atoms.

**Definition 6** *Binding site* is a group of atoms or residues of a protein that usually occurs interaction with drugs or other proteins. The binding site lies in the cavity of protein.

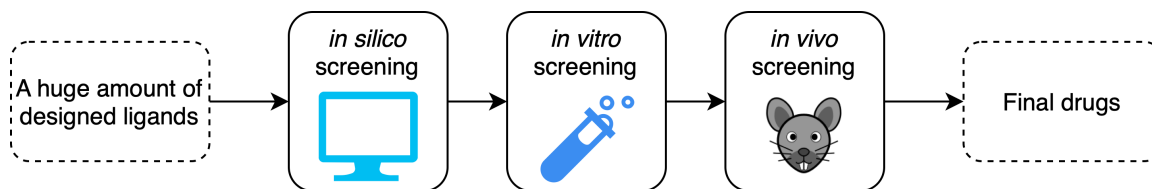
**Definition 7** *Drug design* is a process of building ligand atom-by-atom. This process is also consider as de novo drug design. The drug-builders are expected to add or remove atoms or edges between atoms to create ligands that can be used to form a complete drug. The "stronger" the ligand, the better.

Generally, the process of drug design can be splitted into three sub-processes as belows. Figure 2.2 illustrates the overall process of drug design.

1. *in silico* screening: The designed ligands will be verified to remove know-inactive ligands. Then the remainings will be optimized and verified compulsory properties (e.g. ADMET).
2. *in vitro* screening: The passed *in silico* ligands will be synthesized and put together with the target to confirm their effectiveness.



**Figure 2.1:** An example illustrating the concept of "strong" and "weak" ligand



**Figure 2.2:** Overall process of drug design

3. *in vivo* screening: The confirmed effective ligands will be used to create drugs. These drugs will experiment on live things (e.g. white mice, monkeys, human volunteers) before going to the market.

Based on Theorem 1, we can observe that pharmacologists should try a huge amount of ligands to discover the final drug for a specific virus. Due to that, this process is very time-consuming and costly. To resolve this problem, we should take action pro-actively in *in silico* screening process. There are two options to do in *in silico* screening which can help speed up the whole drug designing process. The first is using tools to predict the weak-interaction (known inactive) ligands and remove them. The most common tools in this option is docking tools which is defined in Definition 8. The second approach is to make the designed ligands better. A definition of a better ligand can be found at Definition 10.

**Definition 8** *Docking* is a process of simulating the drug and target in 3D space and their interactions. This process will put the drug close to a part of the target where interactions will likely occur. The error in simulation of available docking tools is still high for unseen drugs.

As mentioned in the Introduction chapter, the goal of this thesis is creating a decision support system for the drug-designing ViDok system. Our system follow the mentioned second approach give drug-designing suggestions for the users to help them create better drugs. Here, a drug-designing suggestion is defined in Definition 9 to clarify what thing our system gives to the users.

**Definition 9** *A drug-designing suggestion* is an option of adding or replacing an atom or a group of atoms in the designed drug.

**Definition 10** *A better drug* is considered a drug with high probability of being real-life-usable. In the scope of ViDok system, we can simplify this definition to a drug with high ranking (docking score).

The concept of implementing our system is mining the interactions between current drugs available on the ViDok system to learn the designing knowledge from the community then using the learned knowledge to make suggestions. This concept is mainly based on Theorem 2.

**Theorem 2** *All interactions between a drug and the target receptor are based on chemical constraints and rules. These constraints and rules are generalized to all pairs of drug - target protein.*

## 2.2 Decision Support System in Pharmaceutical Formulation

*Decision support systems* (DSS) are usually computer information processing tools that support decision-making activities in the field of particular interest [6]. They are expected to act like an expert with deep knowledge about current problem and ability to give some useful recommendations. The common tasks of a DSS are gathering and analyzing data, synthesizing it to produce comprehensive information reports. These tasks make DSS different from an ordinary operation application [6].

Many years ago, humans only expect the DSS to produce better and better suggestions. In recent years, not only the good suggestions but also explanations for suggestions are made important. Therefore, nowadays, all DSS consists of three basic elements listed in the following.

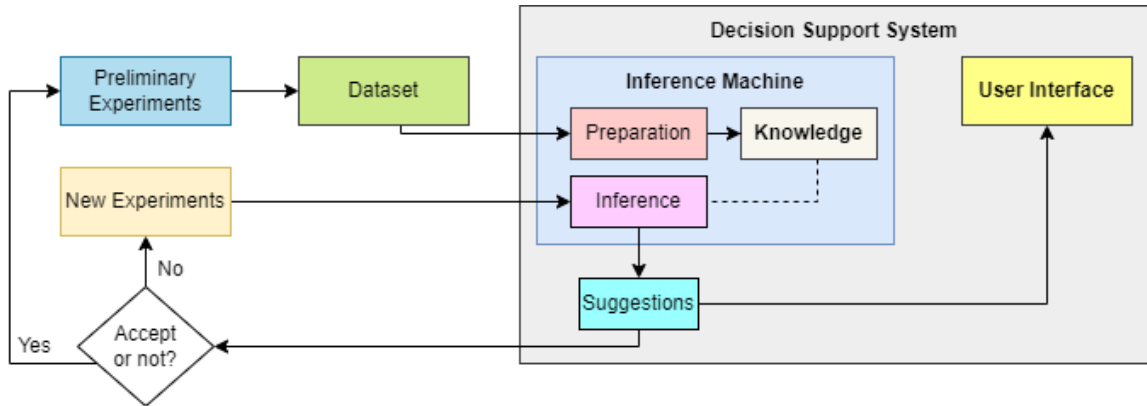
- Knowledge base
- Inference machine
- User interface

A knowledge base consists of all available information that is represented in some convenient ways for future queries. The tasks of collecting and preprocessing knowledge can be very complicated but are crucial for the final system accuracy. In pharmaceutical technology, there are a lot of strong and complex physicochemical constraints and rules, which enables describing the pharmaceutical formulations in terms of their properties better [6]. However, in the other view, those constraints and rules could sometimes not be defined clearly, which makes many obstacles for humans to classically analyze them.

Due to the above reasons, the inference machine is usually a combination of *Artificial Intelligence* (AI) models with knowledge-driven algorithms in order to give meaningful explanations accompany by suggestions [7]. The user interface is a final part of DSS to be prepared and depending on the particular problem, this part will have different implementations.

An overview of working flows of a typical DSS is presented in Figure 2.3. Firstly, the dataset will be constructed from experiments. Then, knowledge from it will be

extracted by the inference machine. Finally, when being used to predict, its results will be shown on the user interface.



**Figure 2.3:** The overview of a typical decision support system

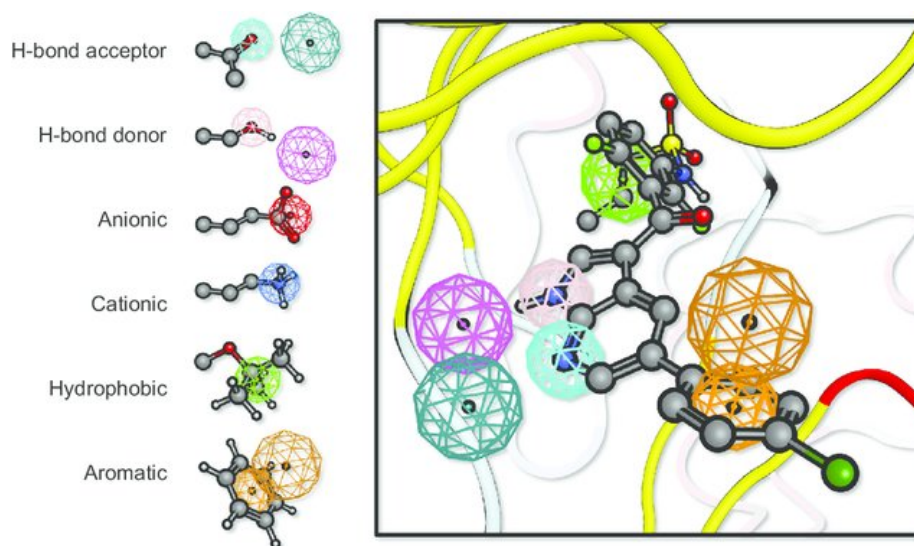
## 2.3 Target Cavity Model for Drug-designing Suggestion

Basically, to create suggestions for building ligand pharmacologically active to a target protein, we base on the properties and attributes of the target protein. According to the Definition 5 of protein cavity, we can consider that protein cavity can be used as the knowledge for drug design decision support system. Therefore, it should be modeled into a representation that contains important features. That representation is called the cavity model, which is defined in Definition 11.

**Definition 11** *Cavity model* is a method that represents all properties of the cavity of a protein. Based on the requirements of the specific tasks, the cavity model can be in many different kinds.

There are many kinds a cavity model can be such as pharmacophore [1], structural interaction fingerprint (SIFt) [8], etc. Among those kinds, the pharmacophore model seems the easiest and most efficient one. The pharmacophore model groups binding atoms into spheres and stores those spheres together with their chemical features. The center of a sphere is the mean point of all atoms creating that sphere while the radius varies as long as larger than the furthest atom belonging to that sphere [1]. An example of pharmacophore model including different sphere types is shown in Figure 2.4.

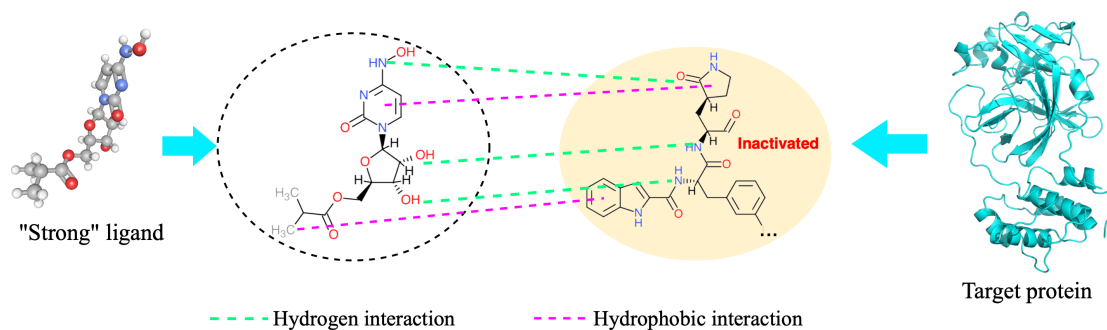




**Figure 2.4:** An example of pharmacophore model containing different features of a protein [1]

## 2.4 Drug-Target Interaction Problem

The *Drug-Target Interaction* (DTI) problem is a common problem in drug discovery and is solved by many methods from traditional to modern [9]. This problem takes a ligand and a target protein as its inputs and uses an algorithm to predict which atoms of the ligand can interact with which atoms of the target protein. A ligand, when being put together with a protein in an appropriate environment, can create many pairwise interactions to that protein. A pairwise interaction can belong to one of three major common types which are *Hydrogen*, *Hydrophobic*, *Van Der Waals* [10]. Each type has different characteristics and its own strengths. Figure 2.5 gives an example of different pairwise interactions created when a ligand interacts with the target protein.

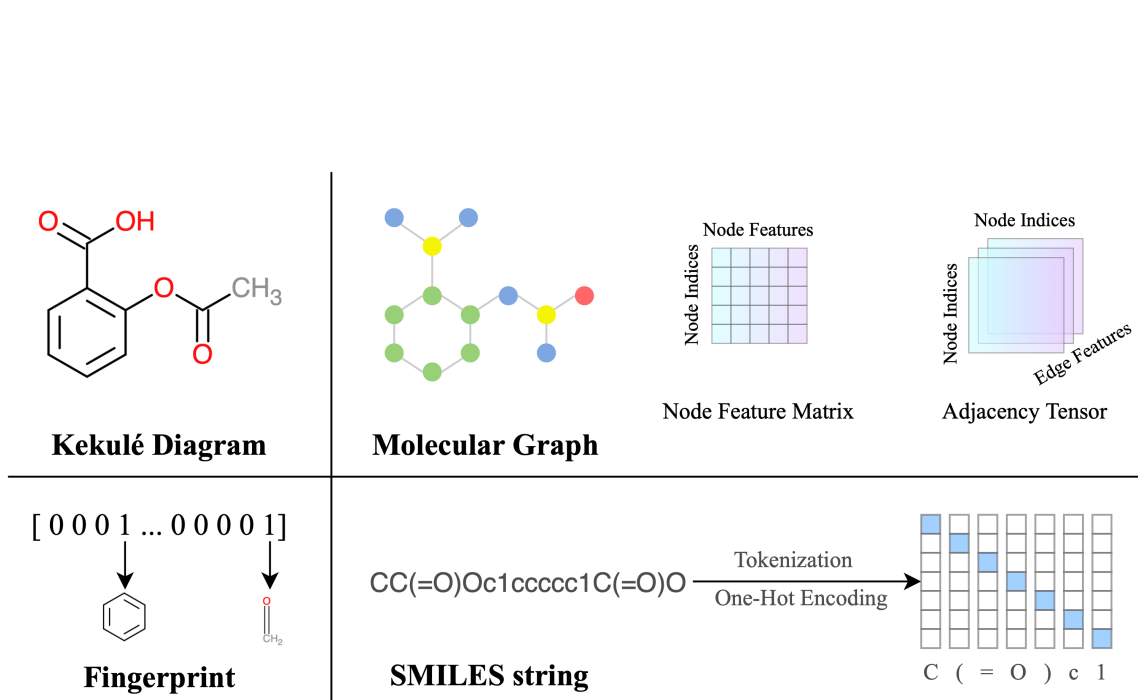


**Figure 2.5:** An example of *Hydrogen* and *Hydrophobic* interactions between a ligand and the target protein

To exactly identify all interactions between a ligand and a protein is an impossible task. The reasons behind this limitation are the variety of ligand 3D structures and the environment where ligand interact with protein and the complex interaction rules. Therefore, currently available methods utilize common chemical rules or combine chemical rules with an AI-based model to provide the prediction of interactions at an acceptable error rate. Further reviews on methods solving this problem can be found in the chapter Related Works below.

## 2.5 Molecular Representation

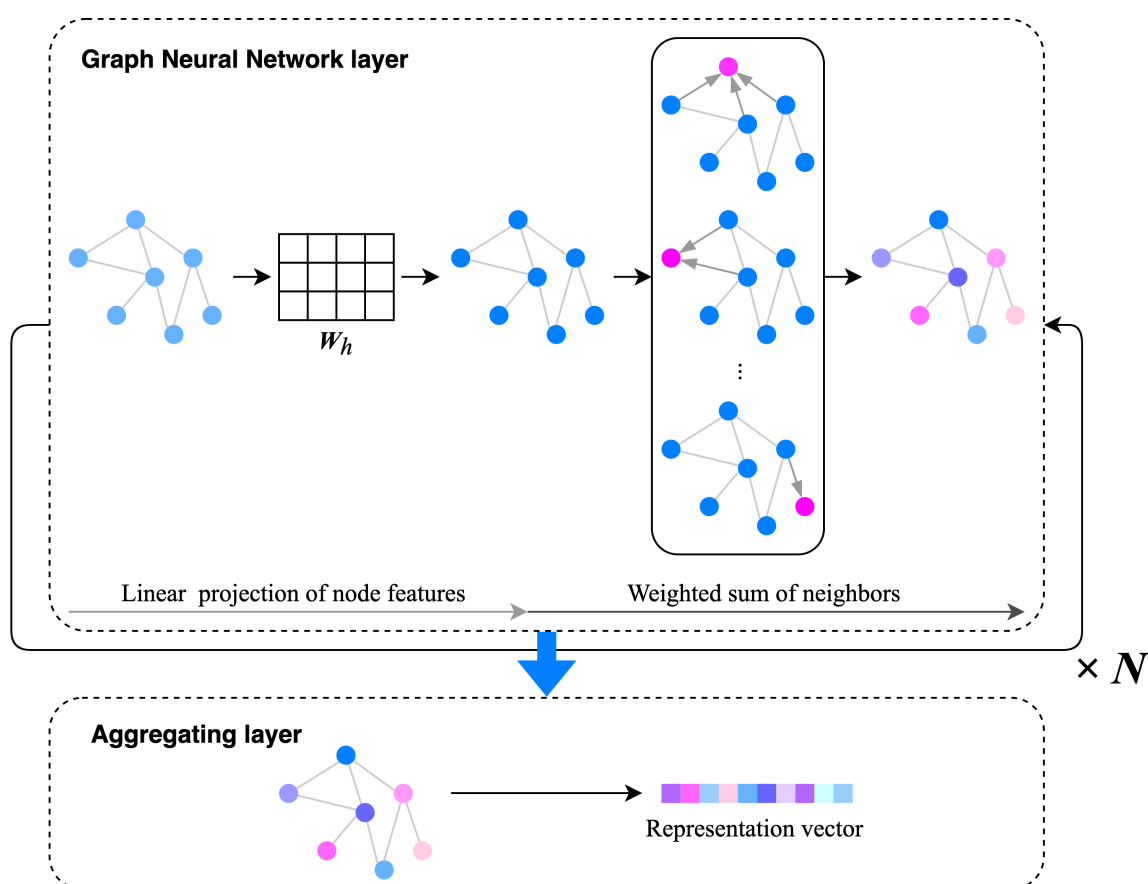
A molecule e.g. a ligand or a protein can be presented in many forms. Some common forms are SMILES string, FASTA string, SMARTS string, feature matrix, image, graph, etc. [11, 12]. Depending on the specific problem and the desire of the authors, proteins and ligands could be in the same or different forms. Using string (SMILE, SMART, FASTA or etc.) as a representation for a ligand or protein is the easiest and best convenient way. But this representation lacks structural information and specific chemical features. This limitation prevents the model to learn robust hidden features so that new tactics to calculate molecular features of protein and ligand (e.g. molecular fingerprint, chemical properties, etc.) have been developed. Some tactics directly calculate features for the whole molecule while others split the molecule into small fragments based on structures or coordinates, calculate features for each fragment then combine them into a matrix [11, 12]. These tactics are good for problems in the virtual screening process that require an overall perspective of a molecule such as toxic prediction or DTI prediction [13]. However, presenting molecules by feature matrixes still can not reflect the true intra-molecular connection (bonding) e.g. the bonding between two atoms. Due to that reason, many recent studies start using graphs as a salvation way to better embed the bonding information. A graph is defined by a set of nodes and edges. When applied to a molecule, it is a set of atoms and bonds. Presented molecules in form of graphs commonly mean that each atom is embedded by a vector and all bonds are embedded by an adjacency matrix [14]. Figure 2.6 gives some examples of common molecular representations including Kekulé diagram (image), SMILES string, fingerprint vector and graph.



**Figure 2.6:** Some examples of common molecular representations

## 2.6 Graph Attention Network

A graph can be defined by  $(V, E)$ , where  $V$  is a set of nodes and  $E$  is a set of edges. The representation of a graph can be an adjacency matrix, adjacency list or incidence matrix. If the graph nodes have attributes, they will be represented by vectors. The *Graph Neural Network* (GNN), which was born in 2009 [15], has been explored and developed in various different domains and has proved its noticeable performance in many applications. Many variants of GNNs are formed to adapt to specific domains. A conceptual view of a GNN layer is illustrated in Figure 2.7.



**Figure 2.7:** Conceptualization of a GNN layer

In general, the GNNs contain two main stages: refining node feature vectors and aggregating all node feature vectors to create the graph feature vector. In the first stage, the feature vector of each node is updated over several times of message passing

between neighboring nodes. The purpose of this stage is to obtain a higher level of feature representation. Then, the refined node feature vectors are aggregated by a function to form the graph feature vector. The aggregating function is required to be invariant over permutations of node ordering. After that, the graph feature will be fed to other layers based on the downstream tasks.

In this thesis, we utilize the power of *Graph Attention Networks* (GAT) [16], which is the most well-known and widely-used variant of GNNs. Assuming we have a graph  $\mathcal{G} = (V, E)$  and each node has a fixed number of features  $F$ , the input for GAT contains an adjacency matrix  $\mathbf{A}$  and a list of node feature vectors  $\mathbf{X}$  defined as (2.1) and (2.2).

$$\mathbf{A}_{ij} = \begin{cases} 1 & \text{if } i \text{ and } j \text{ are connected by an edge} \\ 0 & \text{otherwise} \end{cases} \quad (2.1)$$

$$\mathbf{X} = \{x_1, x_2, \dots, x_{|V|}\} \text{ with } x_i \in \mathbb{R}^F \quad (2.2)$$

Firstly, the GAT performs linear projection to put the node feature vectors into  $F'$ -dimensioned embedding space by (2.3). In (2.3),  $\mathbf{W}_h \in \mathbb{R}^{F' \times F}$  is a learnable weight matrix, and  $x_i^h$  is the projected  $i$ -th node feature vector.

$$x_i^h = \mathbf{W}_h x_i, \quad i = \overline{1, |V|} \quad (2.3)$$

Then the GAT calculates the attention coefficients  $e_{ij}$  for all pairs of  $(i, j)$  nodes. These coefficients are then normalized by the *softmax* function to decrease the bias and the cost of computing. When normalizing for  $e_{ij}$ , only nodes which are connected to  $i$ -th node are considered. Finally, the higher representation of each node is produced by weighted sum of its neighbor nodes using attention coefficients. Equation (2.4a), (2.4b) and (2.4c) are formal definitions of above operations.



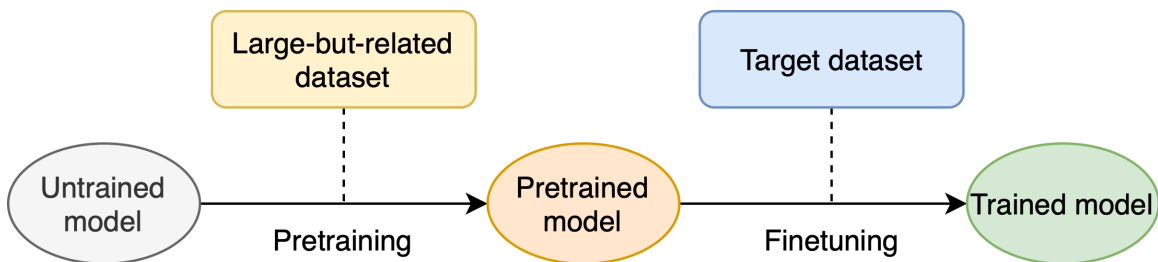
$$\begin{cases} e_{ij} = (x_i^h)^T \mathbf{W}_a x_j^h + (x_j^h)^T \mathbf{W}_a x_i^h, & i, j = \overline{1, |V|} & (2.4a) \\ a_{ij} = \frac{\exp(e_{ij})}{\sum_{k \in C_i} \exp(e_{ik})} \mathbf{A}_{ij}, & i, j = \overline{1, |V|} & (2.4b) \\ x'_i = \sum_{j \in C_i} a_{ij} x_j^h, & i = \overline{1, |V|} & (2.4c) \end{cases}$$

In (2.4a),  $e_{ij}$  is the attention coefficient reflecting the importance of  $j$ -th atom to  $i$ -th atom and  $\mathbf{W}_a \in \mathbb{R}^{F' \times F'}$  is a learnable weight matrix. In (2.4b),  $a_{ij}$  is the normalized attention coefficient corresponding to  $e_{ij}$  and  $C_i$  is the set of neighbor nodes of  $i$ -th node. In (2.4c),  $x'_i$  is the higher representation of  $i$ -th node feature vector. Then the list  $\mathbf{X}' = \{x'_i \in \mathbb{R}^{F'} | i = \overline{1, |V|}\}$  is the output of the GAT layer.

Finally, after going through multiple GAT layers, all the refined node feature vectors in the output node feature matrix  $\mathbf{X}'$  will be aggregated together to form the final representation vector for the whole graph. The aggregating function usually is *sum*, *average*, *max* or *min* function.

## 2.7 Transfer Learning Strategy

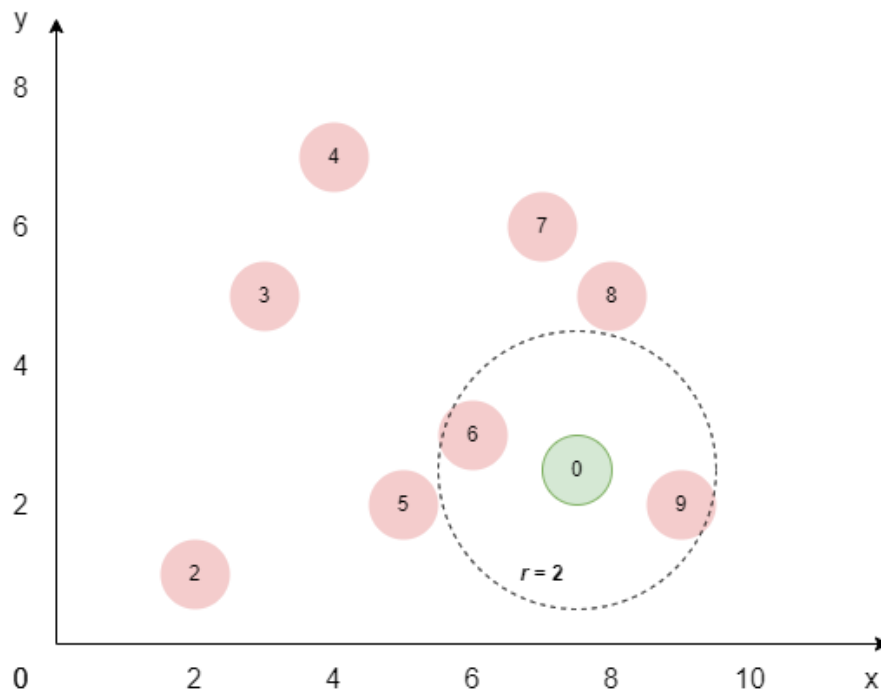
When dealing with small datasets, AI in general and *Deep Learning* (DL), in particular, might not learn effectively, so that many techniques were born to help overcome this case and the most common one is data augmentation [17]. But in some specific cases, the obtained dataset is not large enough for augmentation to work well (e.g. the drug data for COVID-19 disease) or is constrained by tight rules. Thus, we could not perform augmentation on those data but instead, collect more data or use other strategies. In this situation, the idea of using transfer learning becomes more appropriate because it saves time and costs of collecting more data. The definition of transfer learning is written in [18] as follows ”*Transfer learning and domain adaptation refer to the situation where what has been learned in one setting is exploited to improve generalization in another setting.*”. Based on the concept of transfer learning, to solve the problem of small datasets without collecting more data, we can use another large-but-related dataset to pretrain the model and then finetune the pre-trained weights on the target dataset. This process will help the model generalize the learned patterns better, thus, boosting the model performance on the target task. Figure 2.8 visualizes the described pipeline to deal with small datasets using transfer learning.



**Figure 2.8:** An overview of transfer learning strategy

## 2.8 Nearest Neighbors

*Nearest Neighbors* (NN) [19] is a well-known machine learning algorithm for solving many practical problems from easy to complex. The principle behind NN methods is to find a predefined number of available points closest in distance to or in a distance range of the new point. The distance can, in general, be any metric measure and standard Euclidean distance is the most common choice. An example of using this NN method is illustrated in Figure 2.9.



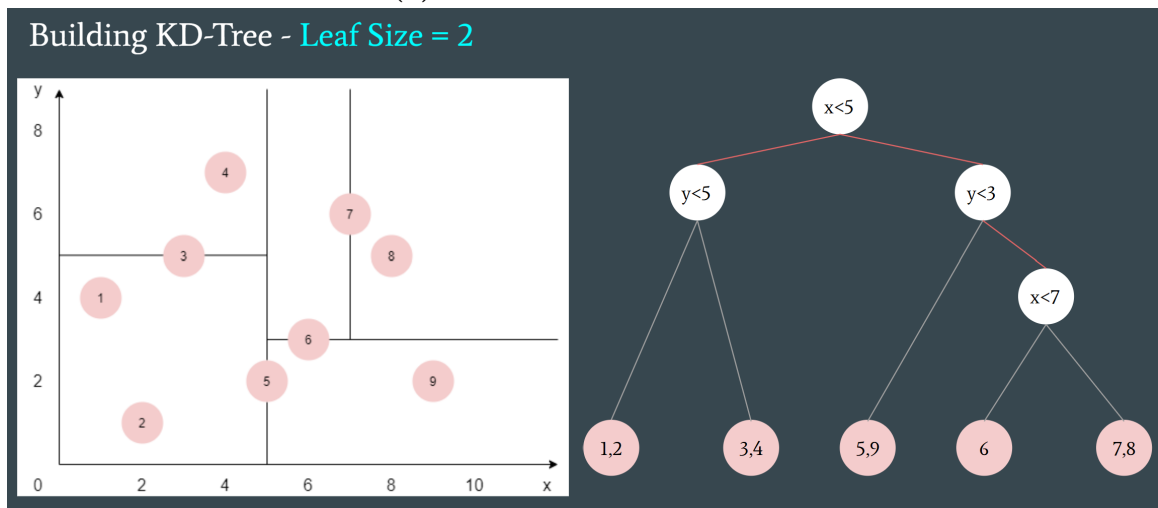
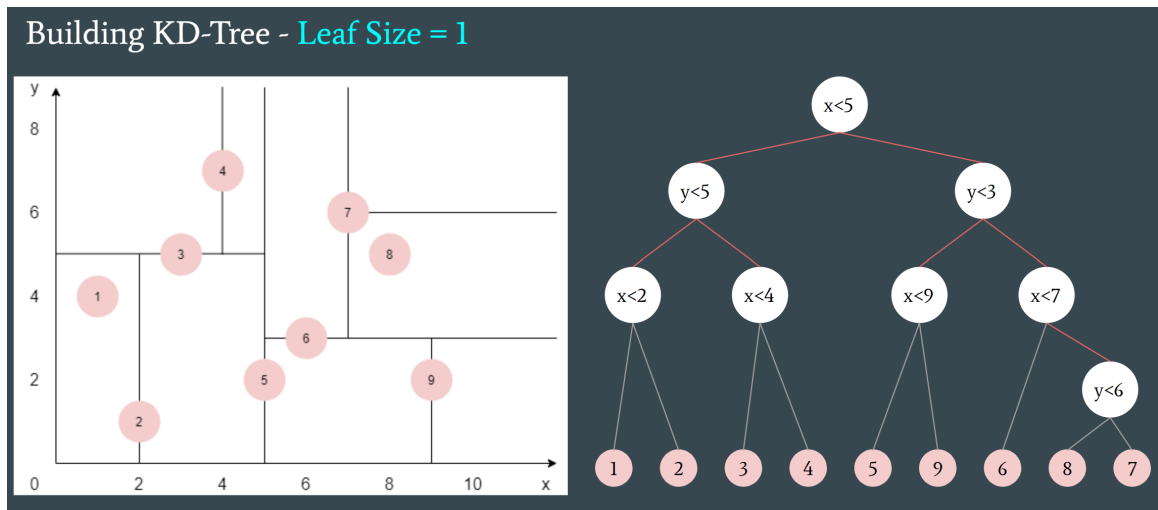
**Figure 2.9:** An example of using Nearest Neighbors algorithm. The available points are colored as red while the new point is colored as green and the cut-off distance equals 2.

In original NN method, the computing complexity is high because the NN algorithm need to calculate all distances to all available points when a new point being added [20]. To reduce this limitation, all available points are used to form a *BallTree* [21] or a *KDTree* [22]. The most common way is using *KDTree*. In *KDTree*, all points will be split into branches and leaves. An example of building a *KDTree* with

different stop conditions is shown in Figure 2.10. The process of building KDTree contains steps as following.

1. Random choose a feature
2. Find the median point of the chosen feature across all data points
3. Split all data points in current node into two branches using the median point
4. Repeat above steps until stop conditions are met. Stop condition can be the deep of the tree or the maximum number of points at leaf nodes.

Despite its simplicity, nearest neighbors has been successful in a large number of classification and regression problems, including handwritten digits and satellite image scenes. Being a non-parametric method, it is often successful in classification situations where the decision boundary is very irregular.



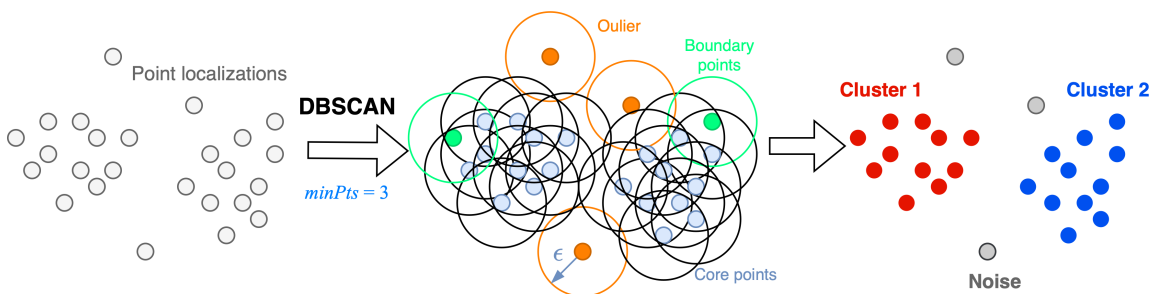
**Figure 2.10:** An example of building KDTree from a dataset with different stop conditions

## 2.9 DBSCAN Algorithm

*Density-Based Spatial Clustering of Applications with Noise* (DBSCAN) is a clustering algorithm which is first proposed by Martin Ester *et al.* in 1996 [23]. It is the most well-known clustering algorithm basing on density. Conceptually, this algorithm groups regions satisfying a density threshold  $minPts$  within a distance of  $\epsilon$  together. In an abstraction way, DBSCAN can be described in five steps as followings.

1. Randomly chose a core point satisfying the condition of density in  $\epsilon$  radius crowder than  $minPts$ .
2. Expand the area of the selected core point by repeating to select other core points in the  $\epsilon$  radius of that area until no further core point can be selected.
3. Assign all points selected to a cluster.
4. Repeat the above three steps until there is no remaining core point.
5. Assign non-selected points to noises.

An example of running DBSCAN on  $minPts = 3$  with defined  $\epsilon$  is illustrated in Figure 2.11.



**Figure 2.11:** An example of DBSCAN clustering two regions

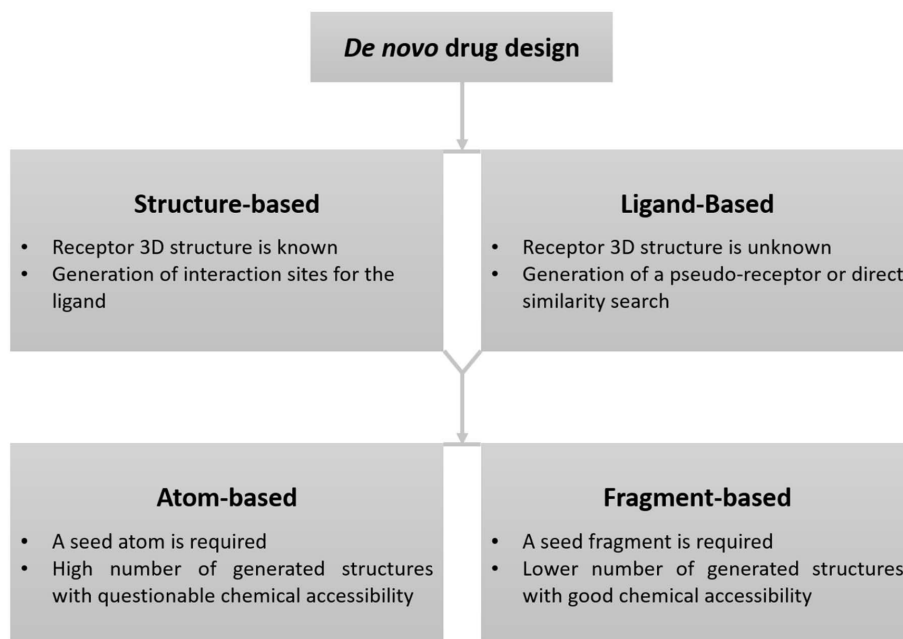
# Chapter 3

## Related Works

### 3.1 *De novo* drug design

*De novo* drug design refers to the design of novel ligands that satisfy a set of chemical constraints and rules using a computational algorithm [24]. The phase "*de novo*" indicates that the ligands are designed from scratch without any starting templates [25]. The process of designing drugs is based only on the information regarding a target protein or its known binding site. Designed drugs are expected to possess good binding or inhibitory activity against the target protein [26, 27, 28]. There are two major approaches available including structure-based and ligand-based design (Figure 3.1) [29]. Both approaches can be implemented at atom level or fragment (a group of atoms) level based on the specific target protein and the choice of the research group.

Toward the structure-based *de novo* drug design, it starts by defining the active binding site of the protein. Then, based on the binding site descriptions, experts can make appropriate decisions. Non-covalent interactions between ligand and protein are mainly in three types, which are *Hydrogen*, *Hydrophobic* and *Van Der Waals*. Since the shape and physicochemical properties of the binding site are crucial for designing a good drug, they should be deeply explored. There are several methods used to identify the interaction site of the protein. Some rule-based methods that can be mentioned are HSITE [30], LUDI and PRO\_LIGAND [31, 32], HIPPO [33],



**Figure 3.1:** A schematic graph representing current *de novo* drug design approaches

etc. These approaches have the same weakness of lacking the ability to detect complex interactions. Therefore, recent researches prefer using knowledge-driven Machine Learning or Deep Learning methods (see [34, 35, 14, 36, 37]). Each method of these has its own advantages and disadvantages so that we should deeply analyze each one before making the decision to use which. After a drug is designed or a drug suggestion is made, it should be evaluated, and this is one of the most important steps in drug design. To evaluate drug candidates, the common way is using a scoring function. There are many types of scoring functions including force fields, empirical scoring functions, and knowledge-based scoring functions, etc. (see [38, 39, 40]).

The other ligand-based approach is only used when the 3D structure presentation of the target protein is unknown. This method uses the active binders from a big dataset to create a ligand pharmacophore model. After that, that ligand pharmacophore model is utilized to create a pseudo-protein then perform similarity search with target protein or directly make drug candidate. The main weakness of ligand-based approaches is limited predictive power when there are insufficient bioactivity data for the ligands towards the target protein. Some research about ligand-based



drug design methods can be listed as DOGS [41], QSAR [42], hybrid QSAR [43], CSP-SAR [44], generative models using Deep Learning [45, 46], etc.

The crowd-sourcing ViDok drug design system follows the concept of the structure-based approach at the atom level in drug design, which makes the process to create a decision support system easier.

## 3.2 Computer-Aided Drug Design Systems

*Computer-aided drug design* (CADD) systems have been appeared more than one decade and no one can deny its influence to the development of several therapeutically crucial ligands [47]. Thanks to those systems, many designed ligands have been successfully developed into commercially available drugs. In the scope of this thesis, we only consider systems used in the *in silico* screening process. Some current highlighted CADD systems used for COVID-19 crowdsourcing drug design are COVID Moonshot [48] and ViDok [4]. The screenshots of these two systems are given in Figure 3.2 and Figure 3.3, respectively. A common CADD system often includes modules such as docking, ligand building suggestions, interactions identification and visualization [47]. In the past, most modules of the CADD system are based on quantum computation and known chemical rules. Recent-year research has demonstrated the power of modern algorithms, especially AI-based ones to overcome the limitations of the past. Some highlight tools widely used in CADD systems can be mentioned such as BIOVIA Discovery Studio [49] for multiple purposes, AutoDock Vina [50] for docking, PyMOL [51] for post-docking analysis, Schrödinger Maestro [52] for molecular modeling, etc.

Toward our decision support system, we expect it to be a part of a big CADD system like ViDok system. Therefore, to develop it, on one hand, we are about to apply *state-of-the-art* Deep Learning methods for building “strong” knowledge to ensure that provided suggestions are good enough that the designed ligands have a higher probability to go to laboratory experiments and become one of the effective drugs used in real life. On the other hand, we try to maintain chemical constraints and rules as much as possible to decrease the false-positive candidates and shorten the time of screening processes.

The screenshot shows the COVID Moonshot system interface on the PostEra platform. At the top, the PostEra logo is on the left, and navigation links for 'Submit', 'Submissions', 'About', and 'Discuss' are on the right. The main header features the 'COVID Moonshot' title with a crescent moon icon and a 'Read About the Story' button. Below this is a 'CONTRIBUTE YOUR DESIGNS' section with three columns: 'We will prioritize compounds and send them out for synthesis and testing.' with a 'Methodology' button; 'Track the status of previously submitted molecules.' with a 'Compound Tracker' button; and 'Join the discussion with scientists around the world on our forum.' with a 'Discuss' button. The main content area is titled 'Draw or enter SMILES (add multiple by pressing "Add" after each entry)'. A yellow warning box states: 'Warning: Structural alerts found (see below). Note: these are just warnings, you can still submit the molecule.' Below this is a text input field containing the SMILES string: CC(C)C(=O)OC[C@H]1O[C@@H](n2ccc(NO)nc2=O)[C@H](O)[C@@H]1O and an 'Add' button. A chemical drawing toolbar is visible, and the main canvas displays a 3D ball-and-stick model of the molecule. The model shows a pyridine ring with a hydroxyl group and a carbonyl group, connected to a ribose sugar, which is further linked to a branched aliphatic chain. A legend on the right side of the canvas lists elements: H (white), C (grey), N (blue), O (red), S (yellow), P (orange), F (green), Cl (light green), Br (brown), and I (purple).

**Figure 3.2:** The screenshot of the COVID Moonshot system

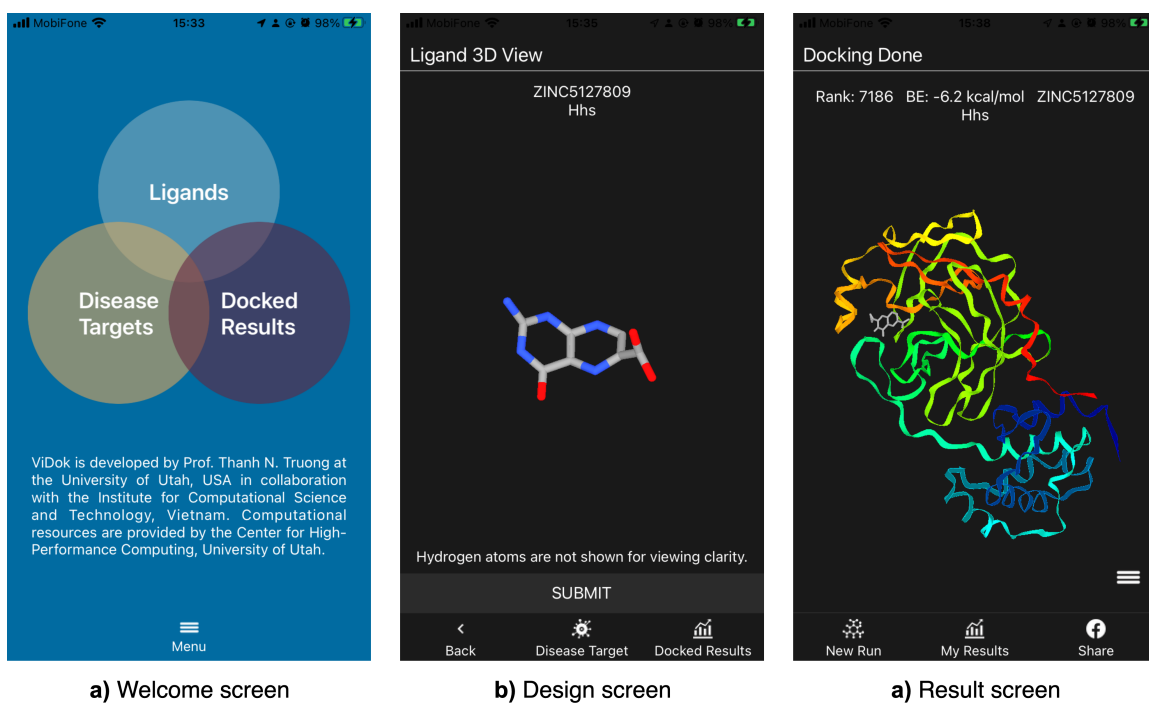
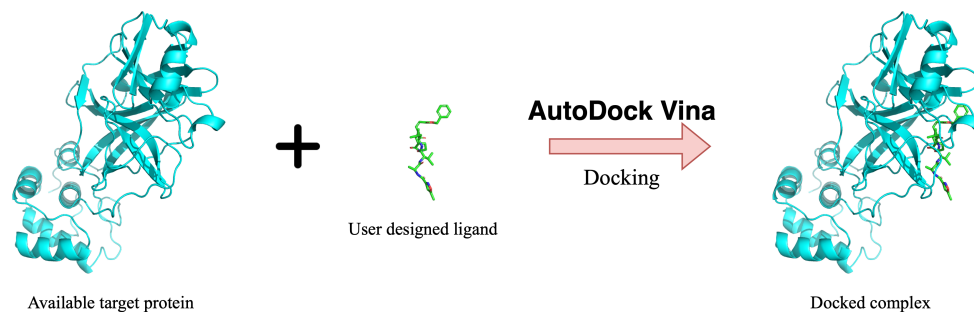


Figure 3.3: The screenshots of the Vidok system

### 3.3 Docking Tools

Docking tools are the most commonly used in the *in silico* screening process. These tools take input of 3D structures of the target protein and design ligand, perform quantum chemical computations and provide the prediction of protein-ligand interaction poses. There are various docking tools such as AutoDock 4 [53], AutoDock Vina [50], GLIDE [54], GOLD [55], LiganScout [56], Molecular Operating Environment (MOE) [57], etc. Among these tools, AutoDock Vina is the one that is most widely used [47, 58]. Despite having been developed for many years, the performance of docking tools in general and AutoDock Vina in particular remains sustainable. The highest performance docking tool is GOLD, which has about 60% of docked ligands having *Root Mean Square Deviation* (RMSD) of atomic positions compared to experiments lower than 2Å [58]. That of the AutoDock Vina is only about 50% [58], which means we can not use these docking tools to analyze or select the effective drugs against any diseases instead removing known weak ligands (ligands locate very far from the common interaction areas of protein). Figure 3.4 visualizes an example of the docking process. Initially, we already have the target protein. After users finish designing a ligand, a docking tool will be called to predict the pose of the ligand when interacting with the protein. These results were visualized using PyMOL.



**Figure 3.4:** An example of docking process

## 3.4 Drug-Target Interaction Prediction

*Drug-Target Interaction* (DTI) prediction is a typical problem in structure-based drug design. A method of solving this problem is expected to take the input of the designed ligand and target protein and analyze pairwise interactions between them. Traditional methods use docked ligand as input and try to check physicochemical rules to consider interactions. But, as discussed in the section Pharmaceutical Knowledge, the chemical rules vary in different environments, thus, it is very hard to define exact rules. As a consequence, the performance of these traditional methods is not high [59].

Recent years show the vast developments of the AI Era in many distinct industries such as health, telecommunication, service industry, etc. Especially in the COVID-19 situation, many AI techniques from *Machine Learning* (ML) to *Deep Learning* (DL) such as *Support Vector Machine* (SVM), Ensemble Learning, *Convolutional Neural Networks* (CNNs), Transformer, *Graph Neural Networks* (GNNs), etc. were applied to help speed up the process of making vaccines and drugs [60] and it makes no exception for the Drug-Target Interaction prediction problem [12]. Currently, AI-based methods can predict pairs of atoms which have high interaction probability then verify the predicted pairs with chemical rules. With this hybrid approach, many hard identifying interaction could be discovered, which makes benefits for the next phase of drug design [59].

At the conceptualization level, almost recent DTI models initially use one suitable AI technique to encode the ligand and protein into feature vectors based on their representations. Then, these models aggregate the two feature vectors together to feed into a classifier to predict if the ligand is pharmacologically active with the protein or not [14, 36, 61, 62, 63]. Some models also utilize the attention mechanism to enhance the performance when creating feature vectors [14, 36, 61] and use attention score to predict high interaction probability atoms.

Another approach for this DTI problem is based on the “guilt-by-association” principle [64, 13]. Methods belonging to this approach try to find similar proteins and ligands in the known-interaction datasets and then use them as evidence to predict the input ligand-protein [64, 9]. This approach has the main weakness when

dealing with strange proteins or ligands. In this case, there is not much "evidence" for guiding the model, so its performance could not be stable [9].

Comparing the above approaches, we can consider that with the unseen protein of Coronavirus, the second approach using similarities will not work as well as the first one with a proper training strategy. Therefore, we are going to follow the first approach (encoding the ligand and target protein then using the produced feature vector to predict interactions).

# Chapter 4

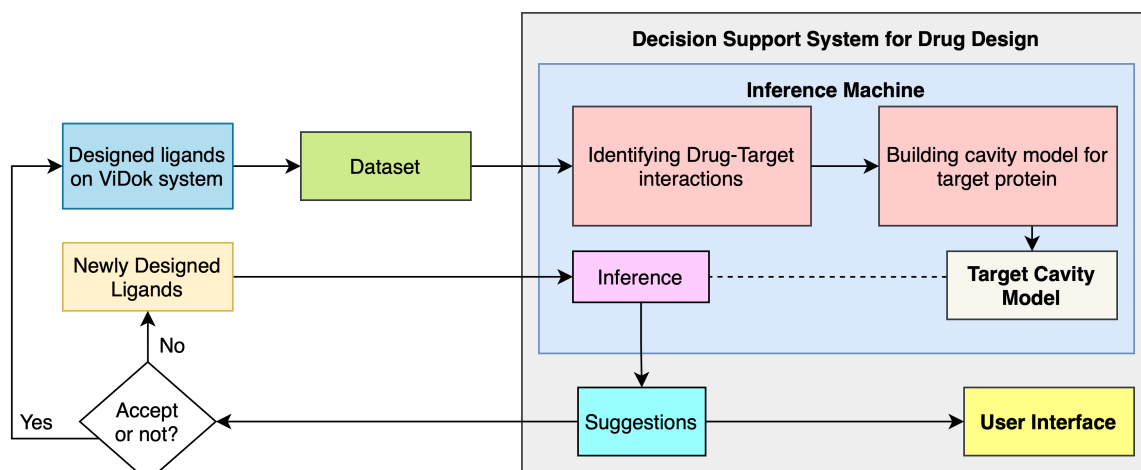
## The Proposed Solutions

### 4.1 Overview

Towards our system, we are going to build it for assisting drug design process. Following the typical decision support system, we adapt its working flow for our system by specifying the input experiments, knowledge and preparation process. Figure 4.2 presents our adapted working flow. In our system, the core knowledge required for creating suggestions is the cavity model of the target protein, so the preparation process will also be changed to two sub-processes with the goal of building this cavity model. Those two sub-processes will solve two necessary problems: i) Extracting interactions between the design ligands and the target protein and ii) Building cavity model for the target protein by mining important interactions. Moreover, as mentioned before, our system will incorporate with the ViDok system. Therefore, our input experiments will mainly collected from the ViDok system.

To emphasize the cooperation between our system with the ViDok system, we visualize their relationship in Figure 4.2. In our system, we consider the one and only target Mpro protein [65] of Coronavirus as the ViDok system. It is the main protein featured for COVID-19 replication. In general, our system will use the drug data both from the ViDok system and other laboratory experiments to prepare knowledge data. Then, when newly designed ligands come to request supports, our system will use learned knowledge together with pre-defined chemical knowledge to make suggestions and send them back to the ViDok system for visualizing.





**Figure 4.1:** The overview of our system following the typical DSS

The target of the preparation process is the cavity model of the target protein. The definitions of cavity and cavity model can be found at Definition 5 and 11. To achieve that knowledge data, we need to mine all interactions that occur between all input drugs and the target protein, which is also known as binding site (Definition 6). Therefore, in the preparation process, we need to solve the problems of two components, which are “identifying drug-protein interactions” and “building cavity model for target protein”. The detail of these two components will be discussed in the following sections.

After the preparation process is finished, our system will have enough knowledge to make suggestions. The inference process will mainly contain two parts that are “finding near atoms in protein cavity” and “performing recommending using learned knowledge”. To simplify the process, we consider these two parts as “building algorithm for recommending” and its detail is also presented in the next section. Finally, we also make a demonstration website for our decision support system alongside with the ViDok system. Below subsection “Web application” describes the details of our developed application.

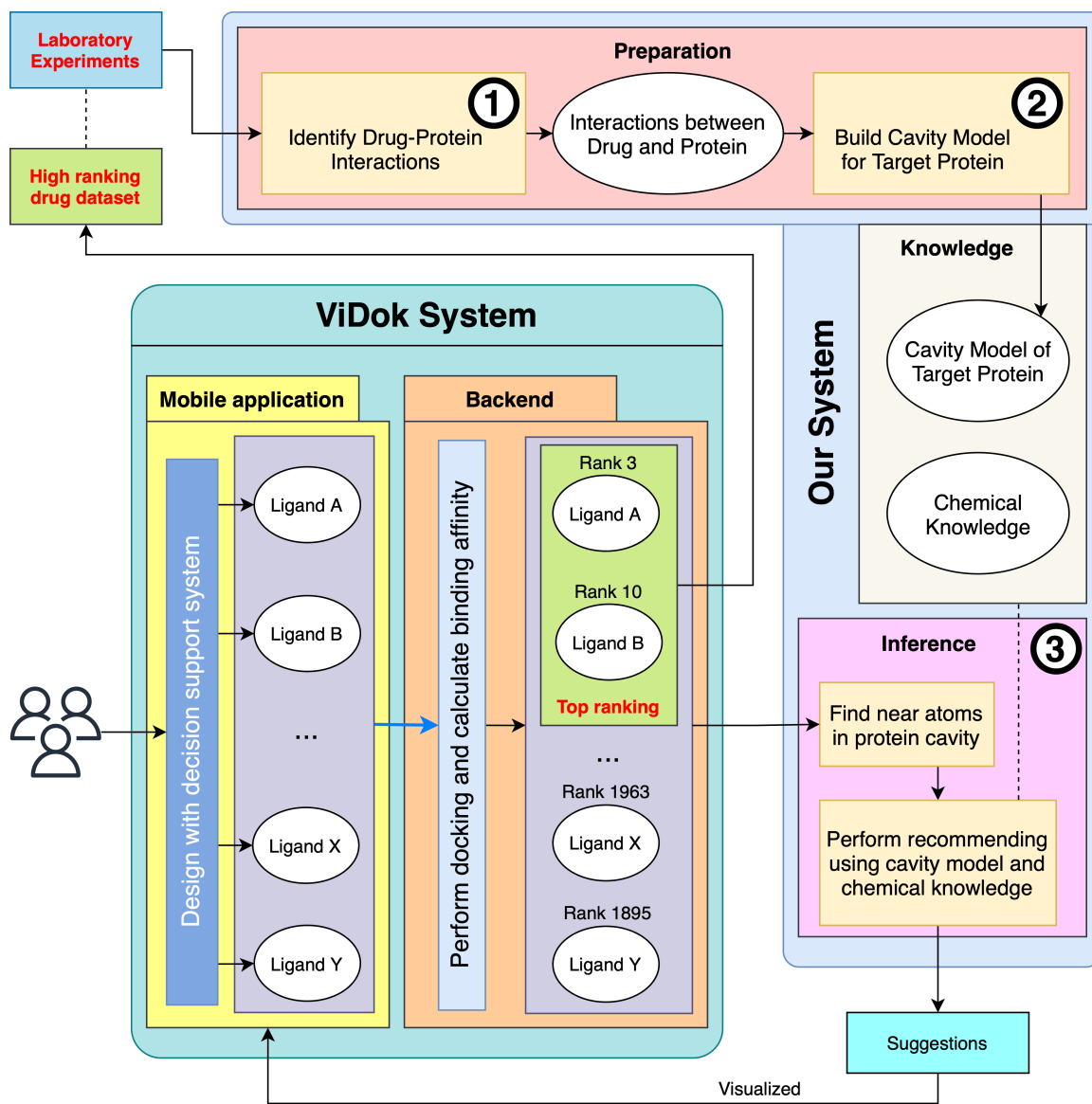
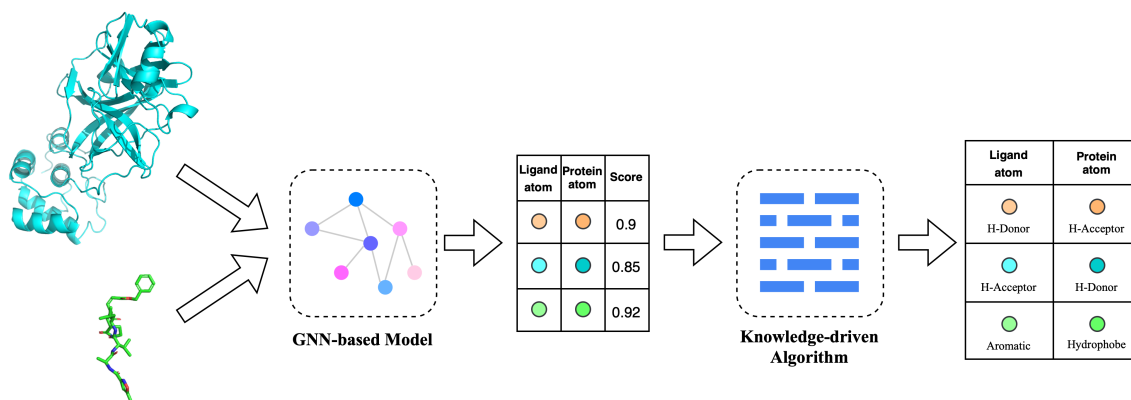


Figure 4.2: The overview of our system together with the ViDok system

## 4.2 Detail of components

### 4.2.1 Identifying Drug-Protein Interactions

The main goal of this components is a method for identifying pairwise interactions between a ligand and the target protein. Our proposed method is a combination of a knowledge-driven algorithm and a Deep Learning model. Firstly, we get the ligand and target protein as input. Then, we use a GNN-based Deep Learning model to extract the high interaction probability pairs by inferencing high attention score elements (the detail will be discussed below). Using those extracted pairs, we use a self-designed algorithm based on *Nearest Neighbors* to check physicochemical constraints. The pairs satisfied constraints are considered pairwise interactions. In the scope of this thesis, we only consider two major interaction types, which are *Hydrogen* and *Hydrophobic*. Figure 4.3 illustrates the overview pipeline of this components.



**Figure 4.3:** The overall pipeline of Identifying Drug-Protein Interactions component

## The Graph Neural Network Model for Identifying High Probability Interactions

At current time, there is no available dataset with fully annotated interactions. Therefore, we perform training model on pharmacologically active predicting task. After model trained, we infer the pairwise interactions by their associate attention scores.

Among state-of-the-art GNN-based model for pharmacologically active predicting task, perhaps the study presented by Lim *et al.* [14] is most remarkable as it does not only apply GNNs but also embeds the 3D structure of the ligand and protein into graph features. This work will then serve as the baseline in my thesis.

Moreover, in the fight against COVID-19, there are not many ligands that have been put in *in vitro* screening. The largest crowd-sourcing drug discovery system for COVID-19, PostEra COVID Moonshot [48], has experimented with only about 2000 ligands. In those experimented ligands, only 379 ligands have been captured co-crystal data i.e. the 3D structure of ligands by Fragalys [66]. As a consequence, current drug-for-COVID datasets lack generalization for training AI models, making them suffering from *overfitting*. We also address this issue by proposing a transfer learning strategy in this thesis. Thus, our contributions are of three-fold as follows.

- Firstly, we apply transfer learning to pretrain the baseline model on a suitable known-interaction variant dataset of the large and diverse dataset from Protein Data Bank [67]. Data from Protein Data Bank is collected from real experiments of pharmaceutical colleagues. Then we finetune the pretrained model for predicting interaction on COVID-19 target protein using the dataset of Fragalys.
- Secondly, we add a mechanism to predict pairwise interactions by inferring attention scores.
- Thirdly, and importantly, we introduce a new model, known as *Atom-enhanced Graph Neural Network with Multi-hop Gating Mechanism*. This model is extended from a baseline model introduced in [14], in which we introduced three major improvement strategies as follows: (i) *Enriched atom encoding*: we additionally encode atoms in the model with more important attributes; (ii) *Total atom aggregation*: we enhance the baseline model to aggregate not only the atoms of ligands but also of the proteins to produce more informative representation of the model output; and (iii) *Multi-hop gating mechanism*: we modify the original gating mechanism to allow non-neighbor atoms affect others, earning improved interaction prediction.

## The baseline model

Our baseline model reuses the study of [14]. This is a novel GNN-based model that can integrate the 3D structure into ligand and protein representations to predict whether the ligand be pharmacologically active with the target protein or not. Figure 4.4 illustrates the overview of the model.

### Model input representation

Firstly, the model takes the input of a ligand and a protein as a graph. According to the original study, while all atoms in the ligand are taken into the graph, only protein atoms in the radius of  $8\text{\AA}$  to any ligand atoms are considered. The graph is presented by a matrix of atom features ( $\mathbf{X}$ ) and two adjacency matrix ( $\mathbf{A}^1$ ,  $\mathbf{A}^2$ ) and equations (4.1), (4.2) and (4.3) shows the way to create these matrixes, respectively. Figure 4.4 has visualized a conceptual view of matrix  $\mathbf{X}$ ,  $\mathbf{A}^1$  and  $\mathbf{A}^2$ .

$$\mathbf{X} = \{x_1, x_2, \dots, x_M\} \text{ with } x_i \in \mathbb{R}^F \quad (4.1)$$

$$\mathbf{A}_{ij}^1 = \begin{cases} 1 & \text{if } i \text{ and } j \text{ are connected by covalent bond or } i = j \\ 0 & \text{otherwise} \end{cases} \quad (4.2)$$

$$\mathbf{A}_{ij}^2 = \begin{cases} \mathbf{A}_{ij}^1 & \text{if } i, j \in \text{protein or } i, j \in \text{ligand} \\ e^{-(d_{ij}-\mu)^2/\sigma} & \text{if } i \in \text{protein and } j \in \text{ligand,} \\ & \text{or if } i \in \text{ligand and } j \in \text{protein} \\ 0 & \text{otherwise} \end{cases} \quad (4.3)$$

In (4.1),  $x_i$  is a feature vector of an atom which contains  $F$  features shown in Table (4.1) and  $M$  is the total atoms in the graph representing both ligand and protein. In (4.2) and (4.3),  $i$  and  $j$  are the atom indexes with the same order as of  $\mathbf{X}$ .  $\mathbf{A}_{ij}^1$  and  $\mathbf{A}_{ij}^2$  are the elements at  $i$ -th row and  $j$ -th column in the  $\mathbf{A}^1$  and  $\mathbf{A}^2$  matrix, corresponding. In (4.3),  $d_{ij}$  is the distance between  $i$ -th atom and  $j$ -th atom and  $\mu$  and  $\sigma$  are learnable parameters.

To adapt this model for our chosen datasets, we have modified the required input by replacing the  $8\text{\AA}$ -radius atoms of protein with the protein pocket (protein cavity).

**Table 4.1:** The list of atom features used in original study and in Improvement 1

Feature	Value
<i>Original</i>	
Atom type	C,N,O,S,F,P,Cl,Br,B,H (onehot)
Degree of atom	0, 1, 2, 3, 4, 5, 6 (onehot)
Number of H atoms attached	0, 1, 2, 3, 4 (onehot)
Implicit valence electrons	0, 1, 2, 3, 4, 5 (onehot)
In aromatic	0 or 1
<i>Added in Improvement 1</i>	
Hydrogen D/A	[is_donor, is_acceptor]
Pos/Neg Ionizable	[is_pos, is_neg]
In lumped hydrophobe	0 or 1

The protein pocket is proved to be the place where almost interactions occur [5]. That means if a ligand is considered active with a protein, it creates many strong interactions with protein atoms in the protein cavity. Therefore, using the protein cavity makes much more sense than the original method.

After all the inputs are prepared, they are then passed to model for predicting ligand-protein interaction.

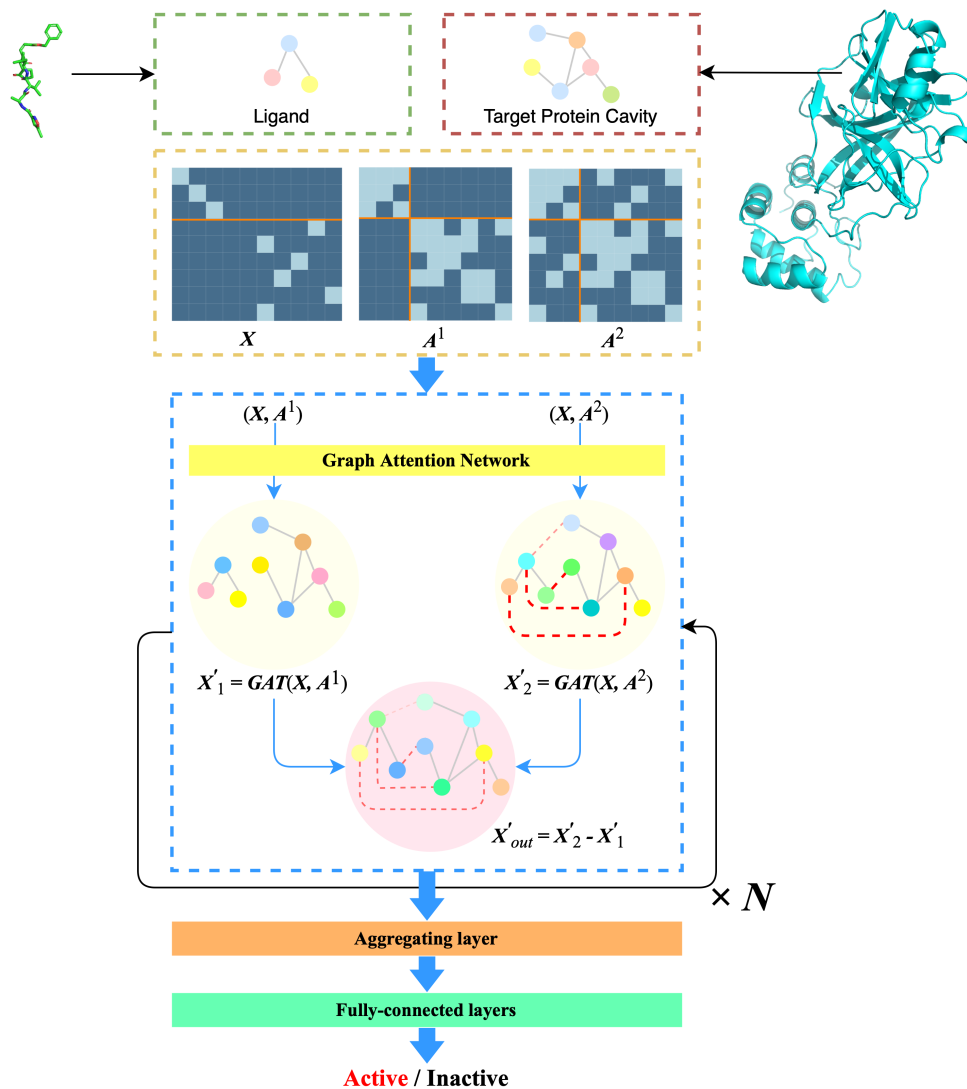
### Model architecture

In this baseline model, Lim and his partners used the GAT layer as the main layer for feature extraction. However, they modified the original GAT layer by adding a gating mechanism at the end of the layer to control how much feature information is passed through. In a formal form, equation (2.4c) of the original GAT is replaced by (4.4).

$$\begin{cases} x_i^{temp} = \sum_{j \in C_i} a_{ij} x_j^h, & i = \overline{1, |V|} \\ z_i = \sigma(\mathbf{W}_o(x_i || x_i^{temp}) + b), & i = \overline{1, |V|} \\ x'_i = z_i x_i + (1 - z_i) x_i^{temp}, & i = \overline{1, |V|} \end{cases} \quad (4.4)$$

In (4.4),  $\mathbf{W}_o \in \mathbb{R}^{1 \times 2F'}$  is a learnable weight matrix and '||' is the concatenation operator.

Let  $\mathbf{GAT}()$  be the formal representation of all GAT layer formulations which are



**Figure 4.4:** The architecture of the baseline model

(2.3), (2.4a), (2.4b) and (4.4). With the input of  $(X, A^1, A^2)$  created from the complex of protein and ligand, we define a GAT block that takes these input and produces the higher representation for  $X$ . Specifically, the GAT block separates the input into  $(X, A^1)$  and  $(X, A^2)$ , passes them to GAT layer to get output  $X'_1$  and  $X'_2$  and perform subtraction  $X'_2 - X'_1$  for model to learn the difference between the structure in a binding pose and the structure as separated. Equation (4.5) presents insights of a GAT block.

$$\begin{cases} \mathbf{X}'_1 = \mathbf{GAT}(\mathbf{X}, \mathbf{A}^1) \\ \mathbf{X}'_2 = \mathbf{GAT}(\mathbf{X}, \mathbf{A}^2) \\ \mathbf{X}'_{out} = \mathbf{X}'_2 - \mathbf{X}'_1 \end{cases} \quad (4.5)$$

In (4.5),  $\mathbf{X}'_{out}$  is the output of GAT block. According to the original study, authors stacked  $N$  GAT block to achieve better feature representations. This can be done by using the output of the previous GAT block  $\mathbf{X}'_{out}$  with two adjacency matrixes  $\mathbf{A}^1$ ,  $\mathbf{A}^2$  as the input for the next GAT block. Please notice that the number of nodes in the GAT layer is equal to the total number of atoms of both protein and ligand ( $|V| = M$ ).

The output refined atom feature vectors of the last GAT block are aggregated in the next step to form a feature vector  $x^{complex}$  representing the complex of the input protein and ligand. Equation (4.6) gives the formulation for creating this vector. Finally, a classifier with multiple fully-connected layers is employed to decide if the input complex is active or not. A fully-connected layer is a non-linear transformation that is defined in (4.7). An overview of the baseline model is also showed in Figure 4.4.

$$x^{complex} = \sum_{i \in ligand} x_i \quad (4.6)$$

$$y = \sigma(\mathbf{W}_c x + b) \quad (4.7)$$

In (4.7),  $x$ ,  $y$  is the input and output fully-connected layer corresponding. The  $\mathbf{W}_c$  is a learnable weight matrix and  $b$  is the bias. Each fully-connected layer in the classifier has its activation function  $\sigma$  as the *ReLU* function except the *sigmoid* function for the final one.

### Adaptation for Pairwise Interactions Prediction

As mentioned before, the origin baseline model is designed for the pharmacologically active predicting task. Therefore, we need to design an algorithm to evaluate attention scores gotten from the trained model. Specifically, when inferring, we also



input the ligand and protein as required. After that, we let the model perform calculations until the last GAT Block. We then get the normalized attention matrix of the GAT layer in the last GAT Block and filter out pairs with attention scores higher than 0.5. Algorithm 1 describes our designed algorithm more formally and Figure 4.5 abstracts the way our algorithm is integrated into the baseline model.

---

**Algorithm 1:** Attention Inference
 

---

**Input** : GNN-based model with attention mechanism  $\mathcal{M}$ ,  
 Input for GNN-based model  $(\mathbf{X}, \mathbf{A}^1, \mathbf{A}^2)$ ,  
 The total number of atoms  $M$ .

**Output:** List of high interaction probability pairs  
 $\mathbf{P} = \{(i, j, s) | i \in \text{ligand} \ \& \ j \in \text{protein} \ \& \ s \geq 0.5\}$

$\mathcal{M}(\mathbf{X}, \mathbf{A}^1, \mathbf{A}^2)$   
 $\text{lastGATBlock} \leftarrow \text{GetLastGATBlock}(\mathcal{M})$   
 $\mathbf{A} = \{a_{ij}\} \leftarrow \text{GetNormalizedAttentionMatrix}(\text{lastGATBlock})$   
 $\mathbf{P} \leftarrow \emptyset$

**for**  $i$  **in**  $\text{Range}(0, M)$  **do**  
 | **for**  $j$  **in**  $\text{Range}(i + 1, M)$  **do**  
 | | **if**  $a_{ij} + a_{ji} \geq 1$  **then**  
 | | |  $\mathbf{P} \leftarrow \mathbf{P} \cup \{(i, j, \frac{a_{ij} + a_{ji}}{2})\}$   
 | | **end**  
 | **end**  
**end**  
**return**  $\mathbf{P}$

---

## The Transfer Learning Strategy

As mentioned in the above sections, to deal with insufficient drug data of Coronavirus’s Mpro protein, we apply a transfer learning strategy for learning the general pattern (general interaction rules) before exploring specific rules of Mpro protein. Firstly, for each model with configured settings (e.g improvements), we perform pre-training using the PDBbind dataset for the model to obtain the generalizability. After that, we finetune the pretrained weights of the model using the Fragalysis dataset. With this process, we expect the finetuned model learned both the common interaction rules of any ligand-protein and the specific rules for Coronavirus protein (Mpro protein). To demonstrate how transfer learning affects the model performance, we

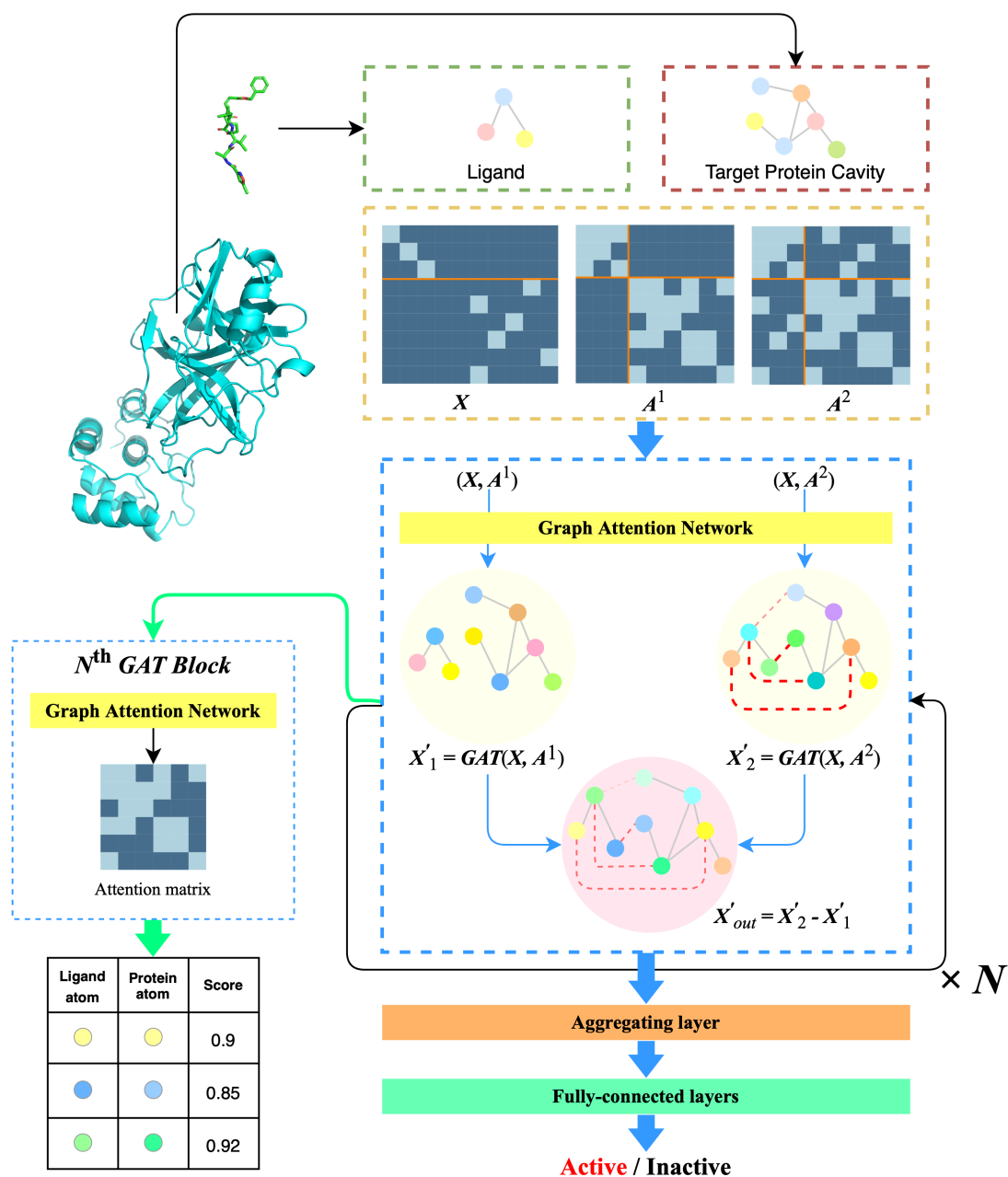
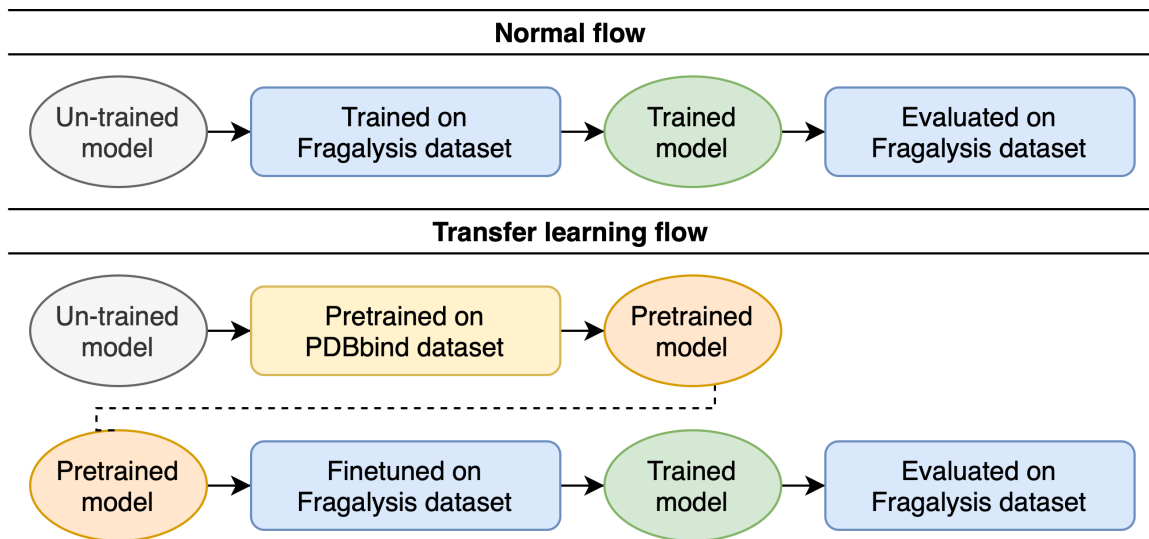


Figure 4.5: The architecture of our modified model

also train the scratch model with the Fragalysis dataset for comparison. Figure 4.6 shows the overview of the above flows including normal flow (model is directly trained

on Fragalysis) and transfer learning flow (model is pretrained before being finetuned on Fragalysis).



**Figure 4.6:** Illustration of the normal flow and transfer learning flow used in this study

## The Atom-enhanced GNN Model with Multi-hop Gating Mechanism

In this study, we introduce an improved model from the baseline one, known as *Atom-enhanced GNN Model with Multi-hop Gating Mechanism*<sup>1</sup>, in which the following improvements are carried out.

**Improvement 1: Enriched atom encoding** The first improvement is enriching atom encoding which adds more chemical features to the representation of each atom. The newly added features include atom degree of six, whether an atom is a hydrogen donor or hydrogen acceptor, whether an atom can be positive or negative ionizable, whether an atom is in any lumped hydrophobe, which are the prerequisites of the corresponding bonding type (a hydrogen bonding requires a hydrogen donor and a hydrogen acceptor atom, so on). Table 4.1 summarizes all features that are used in this improvement.

<sup>1</sup>Our implementation is available at <https://github.com/ViDok-BK/GMGM>

**Improvement 2: Total atom aggregation** The second improvement basically bases on an assumption that interactions are only created if the protein and the ligand match some interaction rules [12]. Therefore, we have modified the original aggregating layer which calculates the sum of all ligand atom vectors to a combination of ligand and protein representation vector. With our modification, any protein atoms that have a minimum distance to any ligand atoms less than  $5.5\text{\AA}$  will be taken into consideration for interaction prediction. The mathematical formulations for our new aggregating layer are proposed in (4.8a)-(4.8c).

$$x^{complex} = (x^{ligand} || x^{protein}) \quad (4.8a)$$

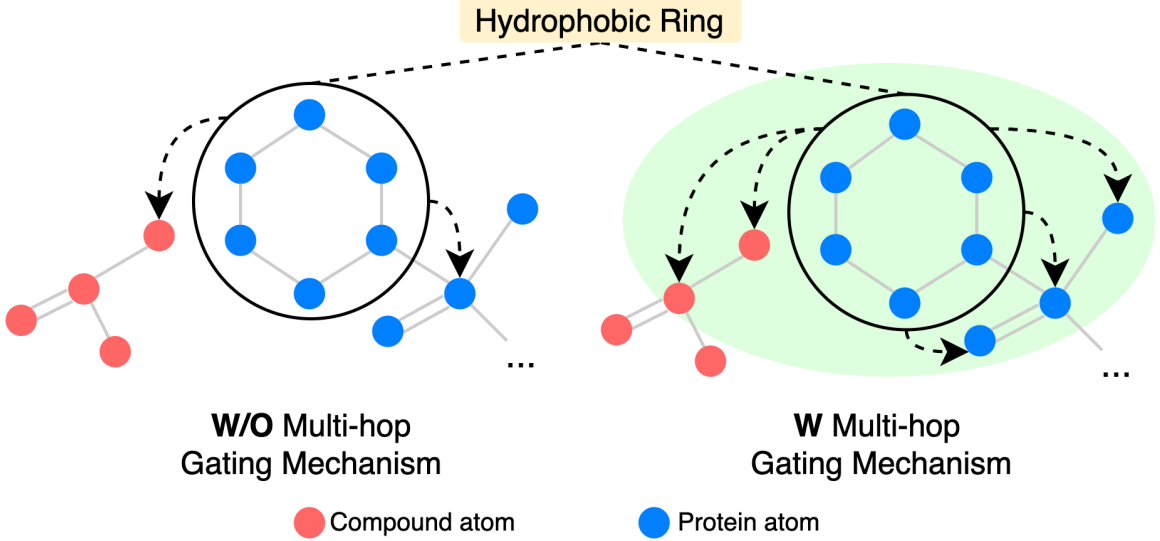
$$x^{ligand} = \sum_{i \in ligand} x_i \quad (4.8b)$$

$$\begin{cases} x^{protein} = \sum_{i \in P} x_i \\ P = \{x_p, p \in protein | \exists c \in ligand : dist(p, c) < 5.5\text{\AA}\} \end{cases} \quad (4.8c)$$

In (4.8a), '||' is the concatenation operator and in (4.8c),  $dist(p, c)$  is the Euclidean distance [68] between protein atom  $p$  and ligand atom  $c$ .

**Improvement 3: Multi-hop gating mechanism** The third improvement is the multi-hop gating mechanism. In this improvement, we repeat the calculation of the gating mechanism multiple times. The reason for this improvement is to enlarge the receptive field of an atom, which is basically based on an assumption in chemistry that atoms having the same function (e.g. hydrophilic, hydrophobic, etc.) usually concentrate together and create a wide area of influence over non-neighbors. Figure 4.7 gives a more intuitive view of this mechanism of influence. This improvement is inspired by the attention diffusion mechanism, which is proposed by [69]. The process of repeating the gating mechanism calculation exactly matches the process of approximate computation for attention diffusion except for the gating coefficient  $z_i$  which is computed from node feature vectors compared to manually input in Wang's study. Let  $K$  be the number of hops in the receptive field of an atom, the process of calculating multi-hop gating mechanism is presented in Algorithm 2. Please notice that in our proposed improvement, we follow Wang's study to use  $x_i^h$  for computing

$x_i^{(k)}$ ,  $k = \overline{1, K}$  instead of  $x_i$  as in original study of Lim.



**Figure 4.7:** The difference between models with and without multi-hop gating mechanism

---

**Algorithm 2:** Multi-hop gating mechanism

---

**Input** : Normalized attention coefficients  $a_{ij}$ , where  $i, j = \overline{1, |V|}$ ,  
 Atom feature vectors  $x_i^h$ , where  $i = \overline{1, |V|}$ ,  
 Number of hops  $K$ .

**Output:** Refined atom feature vectors  $x_i^{(K)}$ , where  $i = \overline{1, |V|}$

$$x_i^{(0)} = x_i^h, \quad i = \overline{1, |V|}$$

**for**  $k$  in  $\text{Range}(1 \dots K)$  **do**

$$\left| \begin{array}{l} x_i^{temp} = \sum_{j \in C_i} a_{ij} x_j^{(k-1)}, \quad i = \overline{1, |V|} \\ z_i = \sigma(\mathbf{W}_o(x_i^{(0)} || x_i^{temp}) + b), \quad i = \overline{1, |V|} \\ x_i^{(k)} = z_i x_i^{(0)} + (1 - z_i) x_i^{temp}, \quad i = \overline{1, |V|} \end{array} \right.$$

**end**

**return**  $\mathbf{X}^{(K)} = \{x_i^{(K)} | i = \overline{1, |V|}\}$

---

## Nearest Neighbors based Algorithm for Verifying Interactions

Having gotten high interaction pairs from previous GNN-based model, we then apply our design algorithm using Nearest Neighbors together with physicochemical rules to verify whether they are actual interactions or not. The chemical rules we used are very basic and satisfied in almost cases. They are presented for two major interaction types as followings.

- **Hydrogen**

- A pair of *Hydrogen Donor* and *Hydrogen Acceptor* (in case ligand atom is *Hydrogen Donor*, protein atom must be *Hydrogen Acceptor* and vice versa)
- Maximum distance between ligand atom and protein atom is 5.5Å

- **Hydrophobic**

- A pair of *Carbon* atom and *Aromatic* ring or a pair of *Hydrophobic* group and *Aromatic* ring
- At least a pair of one ligand atom and protein atom has distance smaller than 5.5Å

The above rules require feature calculation for each atom or group of atoms (*Hydrogen* or *Hydrophobic*). These features can be easily calculated by checking chemical constraints. To ensure the calculation for those features efficient and correct, we use an external library called RDKit [70]. This library is very popular and wide-used by various quantum chemical related projects.

Using the above rules, we develop our algorithm integrating Nearest Neighbors and the results from GNN-based model to get the final pairwise interactions as needed. Our algorithm is presented in Algorithm 3. Figure 4.8 presents a running example using our proposed algorithm.

---

**Algorithm 3:** Nearest Neighbors combined with chemical rules

---

**Input** : High interaction probability pairs

$$\mathbf{P} = \{(i, j, s) | i \in \text{ligand} \ \& \ j \in \text{protein} \ \& \ s \geq 0.5\},$$

ligand, protein,

Distance threshold  $\epsilon_d$ ,Chemical rules  $\mathcal{R}$ .**Output:** Interaction list

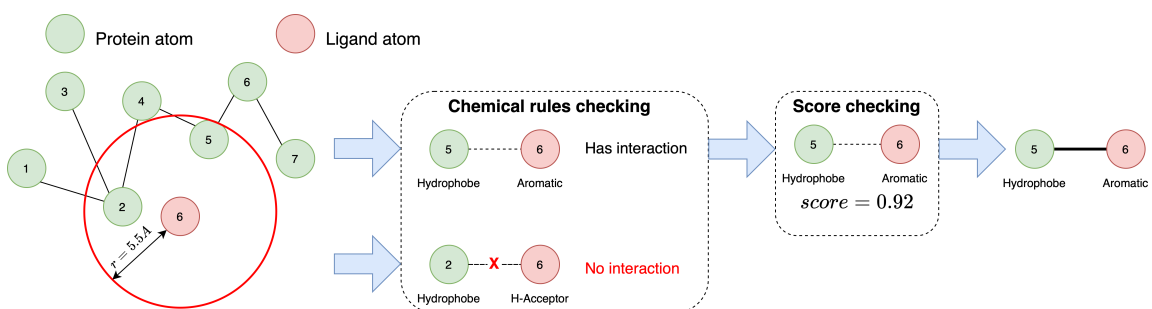
$$\mathbf{I} = \{(i, j, x_i, x_j) | i \in \text{ligand} \ \& \ j \in \text{protein} \ \& \ x \text{ is feature vector}\}$$

 $\mathbf{I} \leftarrow \emptyset$  $\mathcal{N} \leftarrow \text{InitNearestNeighbor}(\text{protein} \rightarrow \text{atoms})$ **for**  $l\_atom$  **in** ligand **do**     $listNearAtoms \leftarrow \text{GetNearAtoms}(\mathcal{N}, \epsilon_d, l\_atom)$     **for**  $p\_atom$  **in**  $listNearAtoms$  **do**         $interaction \leftarrow \text{checkInteractionType}(\mathcal{R}, l\_atom, p\_atom)$         **if**  $interaction$  **is** Hydrogen **then**             $x_i, x_j \leftarrow \text{CalculateFeature}(l\_atom, p\_atom)$              $\mathbf{I} \leftarrow \mathbf{I} \cup \{(l\_atom, p\_atom, x_i, x_j)\}$         **end**        **if**  $interaction$  **is** Hydrophobic **then**             $hasHighProb \leftarrow \text{HasHighProb}(\mathbf{P}, l\_atom, p\_atom)$             **if** not  $hasHighProb$  **then**

| continue

**end**             $x_i, x_j \leftarrow \text{CalculateFeature}(l\_atom, p\_atom)$              $\mathbf{I} \leftarrow \mathbf{I} \cup \{(l\_atom, p\_atom, x_i, x_j)\}$         **end**    **end****end****return**  $\mathbf{I}$ 

---

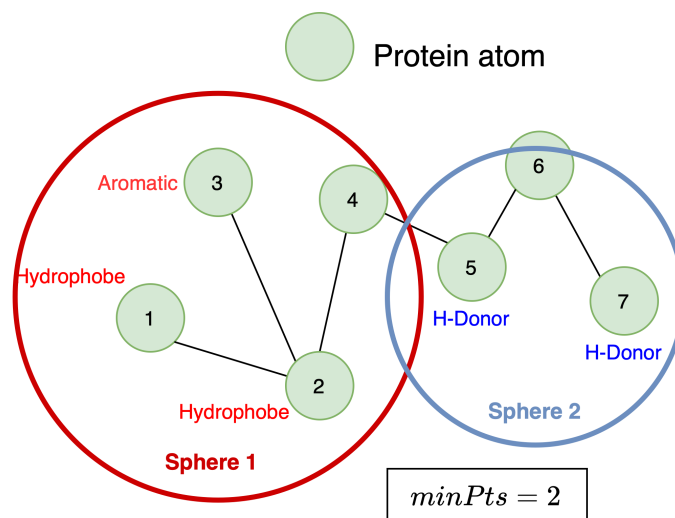


**Figure 4.8:** An example using our proposed Nearest Neighbors based algorithm



### 4.2.2 Building Cavity Model for Target Protein

This component aims to create a cavity model for the target protein to be used in the recommending process. As discussed in Background chapter, the most common cavity model is pharmacophore model which is represented by spheres. Each sphere clusters same feature atoms in a near distance and set the interaction radius. According to chemical knowledge, the more frequent a protein atom has interactions, the more essential it is. Because of that assumption, we need to do an additional task to find frequent interaction protein atoms with the expectation that those atoms will make our suggestions better. With the interactions predicted from the previous component, we put them in 3D space and apply the modified DBSCAN algorithm to cluster those interactions to groups of protein atoms feature by feature. After that, we append a distance of  $5.5\text{\AA}$  to the radius of each sphere as the interaction area. The output of our algorithm is the representation for the protein cavity containing multiple featured spheres. The overview flow is presented in Algorithm 4. An example result is shown in Figure 4.9 to give the conceptual view of a pharmacophore model.



**Figure 4.9:** An example of pharmacophore model flattened in 2D space

---

**Algorithm 4:** Building cavity model for target protein

---

**Input** : The cavity  $\mathcal{C}$  of target protein,  
Interaction list  
 $\mathbf{I} = \{(i, j, x_i, x_j) | i \in \text{ligand} \ \& \ j \in \text{protein} \ \& \ x \text{ is feature vector}\}$ ,  
Minimum number of point  $minPts$ ,  
Radius  $\epsilon$ .

**Output:** Pharmacophore model  $\mathcal{P}$  for the target protein

$\mathcal{P} \leftarrow \emptyset$   
 $pAtomsWithFeature \leftarrow \text{ExtractProteinAtom}(\mathbf{I})$   
 $C \leftarrow 0$   
 $listCluster \leftarrow \emptyset$

**for**  $p\_atom, x_p$  **in**  $pAtomsWithFeature$  **do**  
     $neighbors \leftarrow$   
    RangeQuerySameFeature( $pAtomsWithFeature, p\_atom, \epsilon, x_p$ )  
    **if**  $|neighbors| < minPts$  **then**  
    | continue  
    **end**  
     $C \leftarrow C + 1$   
     $listCluster[C] \leftarrow neighbors$   
    **for**  $n\_atom$  **in**  $listCluster[C]$  **do**  
    | **if**  $BelongAnyCluster(n\_atom)$  **then**  
    | |  $listCluster[C].remove(n\_atom)$   
    | | continue  
    | **end**  
    |  $neighbors \leftarrow$   
    | RangeQuerySameFeature( $pAtomsWithFeature, n\_atom, \epsilon, x_p$ )  
    | **if**  $|neighbors| \geq minPts$  **then**  
    | |  $listCluster[C] \leftarrow listCluster[C] \cup neighbors$   
    | **end**  
    **end**  
**end**

**for**  $cluster$  **in**  $listCluster$  **do**  
     $mean\_point, radius, feature \leftarrow \text{CalculateSphereFeature}(cluster)$   
     $radius \leftarrow radius + 5.5$  // Add the interaction area  
     $\mathcal{P} \leftarrow \mathcal{P} \cup \{(mean\_point, radius, feature)\}$   
**end**

**return**  $\mathcal{P}$ 

---

### 4.2.3 Building Algorithm for Recommending

Conceptually, this module does not try to deeply analyze the interactions but instead quickly scans through the ligand and figure out modification needed atoms to propose suggestions. In this module, we utilize the built pharmacophore model and basic chemical rules as used in the first component of identifying interactions to screen the designed ligand. The expected outputs of this component are suggestions of adding or replacing an atom or a group of atoms.

To prepare for the input for this component, we take the pharmacophore built in the previous component and the chemical rules in the first component. After all the required knowledge is prepared, we now can confidently give suggestions to a newly designed ligand. The overall process of creating suggestions can be grouped into two main steps. The first step is finding all ligand atoms that belong to any sphere in the pharmacophore model. The second step is creating suggestions corresponding to every sphere in pharmacophore model if no ligand atom belongs to that sphere or belonged atoms have unsuitable features. For example, if the sphere has *Hydrogen Donor* feature, but the ligand atom belonged to that sphere does not have *Hydrogen Acceptor* feature, so it is unsuitable. To query the suitable ligand atoms for a sphere, we base on basic chemical rules to predefine suitable atoms for recommending. Our predefined suitable atoms by each rule are listed in Table 4.2. Algorithm 5 summarizes our proposed flow to create suggestions from learned knowledge.

**Table 4.2:** The list suitable atoms for recommending by each chemical rules

Protein feature	Ligand feature	Suitable ligand atom(s)
H-Acceptor	H-Donor	O or O=CO or N or NC=O
H-Donor	H-Acceptor	O=CO or NC=O or C=O
Hydrophobic or C	Aromatic	c1ccccc1 or Cc1ccccc1 or C=Cc1ccccc1
Aromatic	Hydrophobic or C	C

---

**Algorithm 5:** Making suggestions for a new drug

---

**Input** : A newly designed *ligand*,  
Pharmacophore model  $\mathcal{P}$  of target protein,  
Chemical rules  $\mathcal{R}$ .

**Output:** Suggestions  $S$

$S \leftarrow \emptyset$

**for** *sphere* in  $\mathcal{P}$  **do**

$l\_atoms \leftarrow FindAtomInSphere(sphere, ligand)$

**if**  $l\_atoms$  **then**

$satisfied \leftarrow VerifyChemicalRules(\mathcal{R}, sphere, l\_atoms)$

**if** not *satisfied* **then**

$l\_atoms\_ids \leftarrow GetReplaceAtomIndexes(l\_atoms)$

$replace\_atoms \leftarrow QuerySuitableAtoms(sphere, \mathcal{R})$

$s \leftarrow ("replace", l\_atom\_ids, replace\_atoms)$

$S \leftarrow S \cup \{s\}$

**end**

**end**

**else**

$l\_atoms\_ids \leftarrow GetAddAtomIndexes(l\_atoms)$

$add\_atoms \leftarrow QuerySuitableAtoms(sphere, \mathcal{R})$

$s \leftarrow ("add", l\_atom\_ids, add\_atoms)$

$S \leftarrow S \cup \{s\}$

**end**

**end**

**return**  $S$

---

## 4.3 Web application

### 4.3.1 Overview of Application

Before integrating into the ViDok system, we create our separate web application that allows users to design ligands with our decision support system. Our drug design system follows the concept of the ViDok system that takes the user-designed ligands, performs docking computation and returns the docking score (measuring binding affinity of the ligand in *kcal/mol*) together with docked 3D structures. Besides that, our system also gives some suggestions after the docked results are returned to help users quickly figure out where need to change for a better ligand. Toward the architecture of our drug design system, we adopt the desired architecture for ViDok system as shown in Figure 4.2 to our own system using a web application instead of a mobile application. Our adopted architecture is visualized in Figure 4.10.

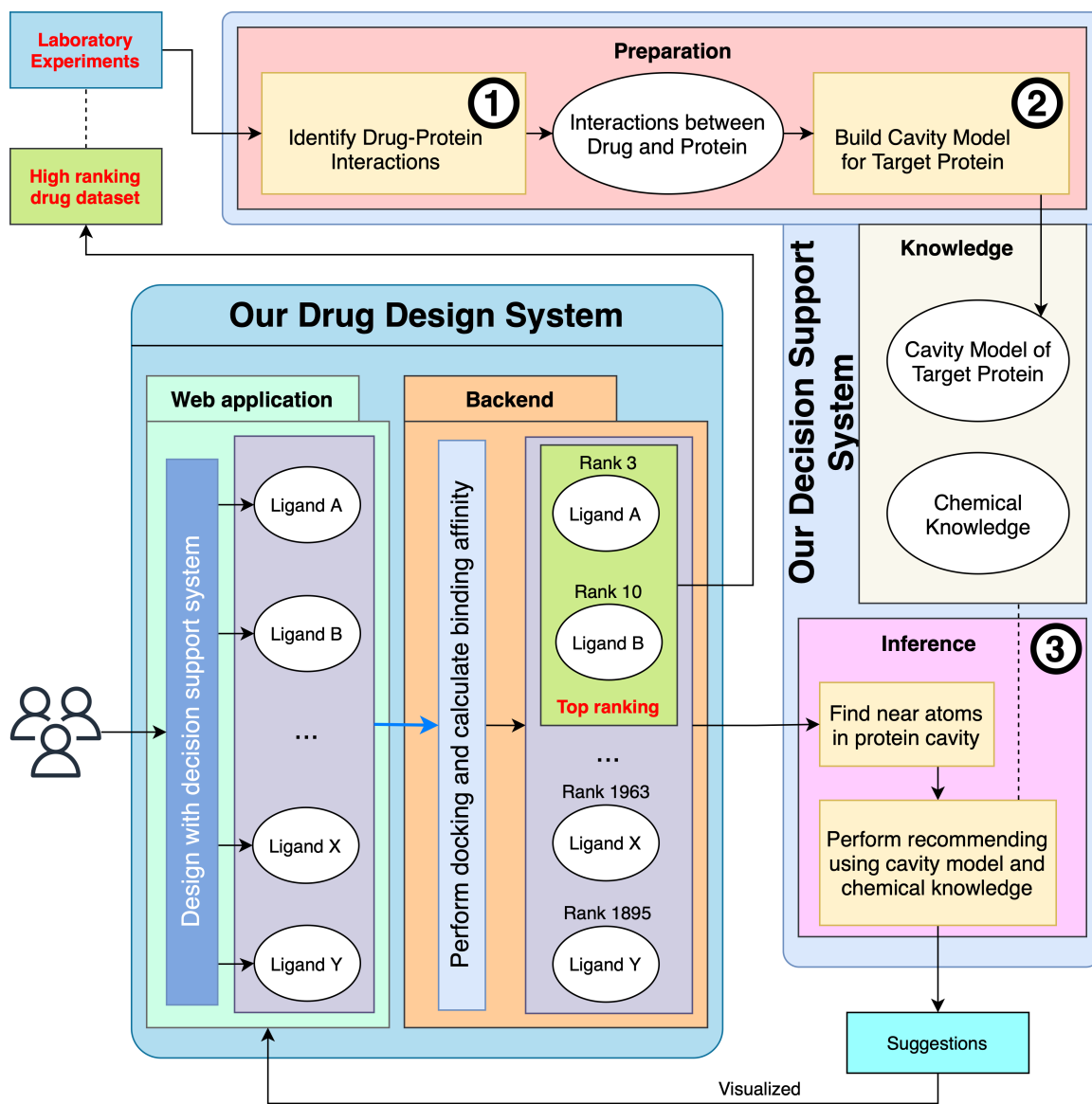


Figure 4.10: Our drug design system integrated AI-powered decision support system

## 4.3.2 Requirements

### Functional Requirements

- The system allows users to design ligand atom by atom, bond by bond.
- The system performs docking and returns the 3D structures with the docking scores to the users after they submit their designs.
- The system only allow users design one ligand at the same time.
- The system provides users suggestions after docking their design ligands.
- User can view their previous designed ligands.
- The system ranks all ligands designed from all users.

### Non-Functional Requirements

- The system can run on multiple platforms such as MacOS, Windows, Linux, iPad OS, Android etc. and on multiple browsers such as Firefox, Chrome and Safari.
- The *User Interface* (UI) of the system can be responsive (rendered without any error).
- Total time for docking and generating suggestions must be below 5 minutes.
- The system can serve upto 10 users at the sametime.

## 4.3.3 Usecase Descriptions

As described above, our system is a drug design system that allows users to perform basic ligand building actions and gives suggestions for their designs. Our system, in general, contains six usecases such as Log in, Log out, Register, Design ligand, Apply suggestion and View designed ligands. Figure 4.11 presents the overview of all usecases in our application. Table 4.3, 4.4, 4.5, 4.6, 4.7 and 4.8 gives the detail descriptions of the mentioned usecases respectively.

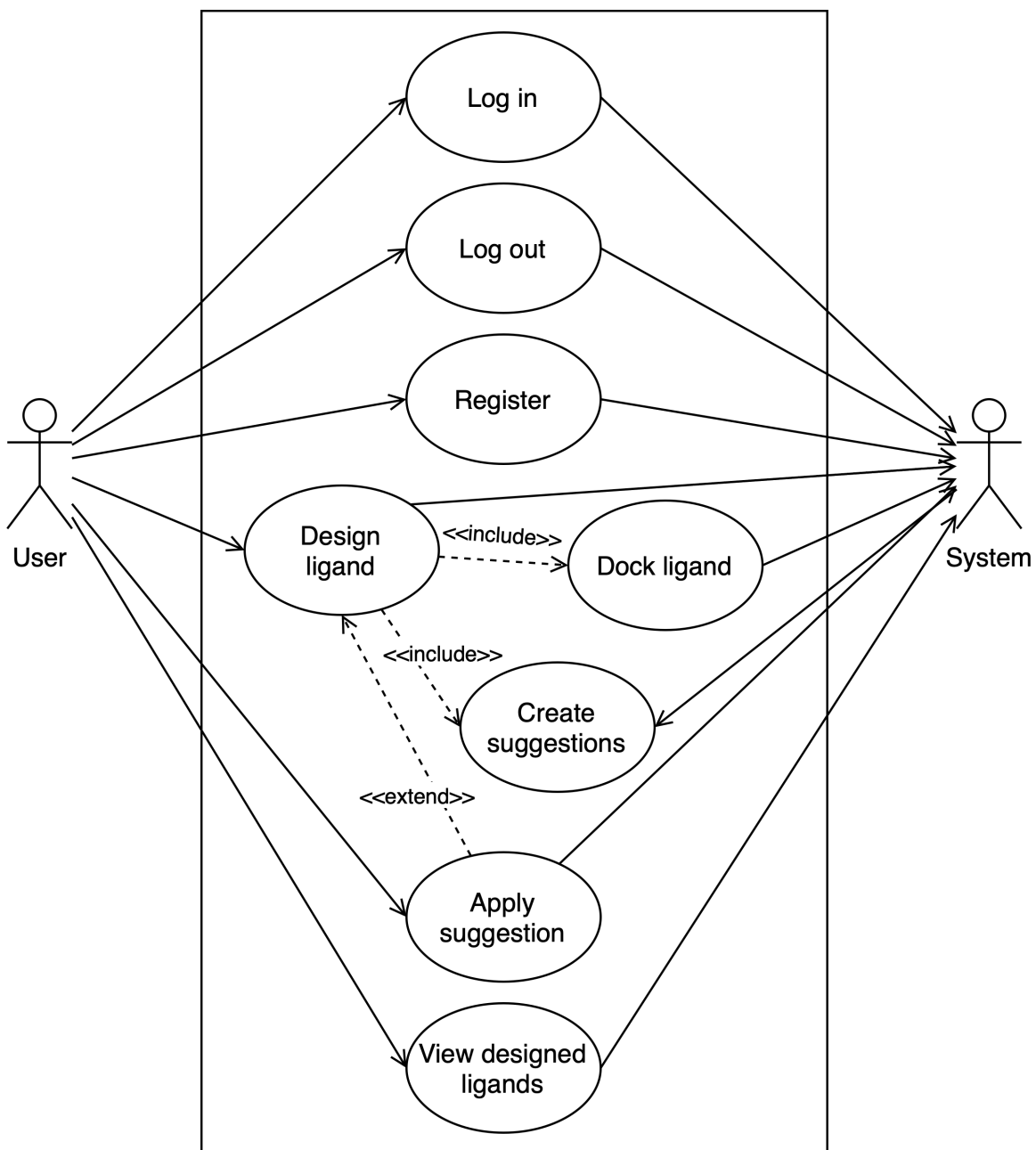


Figure 4.11: Overview of all usecases in our application



**Table 4.3:** "Log in" usecase description

<b>Name</b>	Log in
<b>Actor</b>	User, System
<b>Description</b>	User logs in into the system.
<b>Pre-conditions</b>	User opened the application on an appropriate browser.
<b>Post-conditions</b>	User is logged in.
<b>Normal flow</b>	<ol style="list-style-type: none"> <li>1. User inputs username and password.</li> <li>2. The system verifies credential from the user.</li> <li>3. The system is logged in.</li> </ol>
<b>Exceptions</b>	<i>Exception 1: at step 2</i> 1a. If the system cannot verify user credential, it shows the error message to user.
<b>Alternative flow</b>	There is no alternative flow.

**Table 4.4:** "Log out" usecase description

<b>Name</b>	Log out
<b>Actor</b>	User, System
<b>Description</b>	User logs out the system.
<b>Pre-conditions</b>	User has already logged in.
<b>Post-conditions</b>	User is logged out.
<b>Normal flow</b>	<ol style="list-style-type: none"> <li>1. User presses the Logout button.</li> <li>3. The system is logged out.</li> </ol>
<b>Exceptions</b>	There is no exception.
<b>Alternative flow</b>	There is no alternative flow.

**Table 4.5:** "Register" usecase description

<b>Name</b>	Register
<b>Actor</b>	User, System
<b>Description</b>	User registers a new account with the system.
<b>Pre-conditions</b>	User opened the application on an appropriate browser.
<b>Post-conditions</b>	A new account is created.
<b>Normal flow</b>	<ol style="list-style-type: none"> <li>1. User inputs required information to create account.</li> <li>2. The system verifies that the username and email input by user has not been registered.</li> <li>3. The system creates a new account for user.</li> </ol>
<b>Exceptions</b>	<i>Exception 1: at step 2</i> <ol style="list-style-type: none"> <li>1a. If the username or email has already existed, it shows the error message to user.</li> </ol>
<b>Alternative flow</b>	There is no alternative flow.

**Table 4.6:** "Design ligand" usecase description

<b>Name</b>	Design ligand
<b>Actor</b>	User, System
<b>Description</b>	User design a ligand on the system.
<b>Pre-conditions</b>	User has logged in and opened the design page.
<b>Post-conditions</b>	User receives docking score and 3D structure of ligand-protein complex.
<b>Normal flow</b>	<ol style="list-style-type: none"> <li>1. User draws a new ligand on the system UI.</li> <li>2. User presses Submit button.</li> <li>3. The system performs docking computation.</li> <li>4. The system creates suggestions for the docked ligand.</li> <li>5. The system returns docking score and suggestions to the user.</li> </ol>
<b>Exceptions</b>	<i>Exception 1: at step 2</i> <ol style="list-style-type: none"> <li>1a. If the system cannot perform docking ligand, it shows the error message to user.</li> </ol>
<b>Alternative flow</b>	There is no alternative flow.

**Table 4.7:** "Apply suggestion" usecase description

<b>Name</b>	Apply suggestion
<b>Actor</b>	User, System
<b>Description</b>	User applies a suggestions given by our system.
<b>Pre-conditions</b>	User submitted the designed ligand and received suggestions from our system.
<b>Post-conditions</b>	The suggestion is applied to the current design of the user.
<b>Normal flow</b>	<ol style="list-style-type: none"> <li>1. User selects the suggestion to be applied.</li> <li>2. The system merges the selected suggestion into current design.</li> </ol>
<b>Exceptions</b>	There is no exception.
<b>Alternative flow</b>	There is no alternative flow

**Table 4.8:** "View designed ligands" usecase description

<b>Name</b>	View deisgned ligands
<b>Actor</b>	User, System
<b>Description</b>	User views previous designed ligands.
<b>Pre-conditions</b>	User has already logged in.
<b>Post-conditions</b>	The list of designed ligands are shown on the system UI.
<b>Normal flow</b>	<ol style="list-style-type: none"> <li>1. User presses View ligands button.</li> <li>2. The system shows the designed ligands onto the UI.</li> </ol>
<b>Exceptions</b>	There is no exception.
<b>Alternative flow</b>	There is no alternative flow

### 4.3.4 Database Design

To store the user information and user-designed ligand, we save those data into a database. Toward the schema for the users, it will store the username, full name, email and password attributes. The username attribute is the primary key of the *User* schema. The *Ligand* schema is dependent on the *User* schema and will have three attributes as submit time, saved path and docking score. In those attributes, the attribute of submitting time is the partial key of the *Ligand* schema. Figure 4.12 presents the *Enhanced Entity-Relationship Model* (EERD) of our designed database and Figure 4.13 visualizes the detailed design of the database.

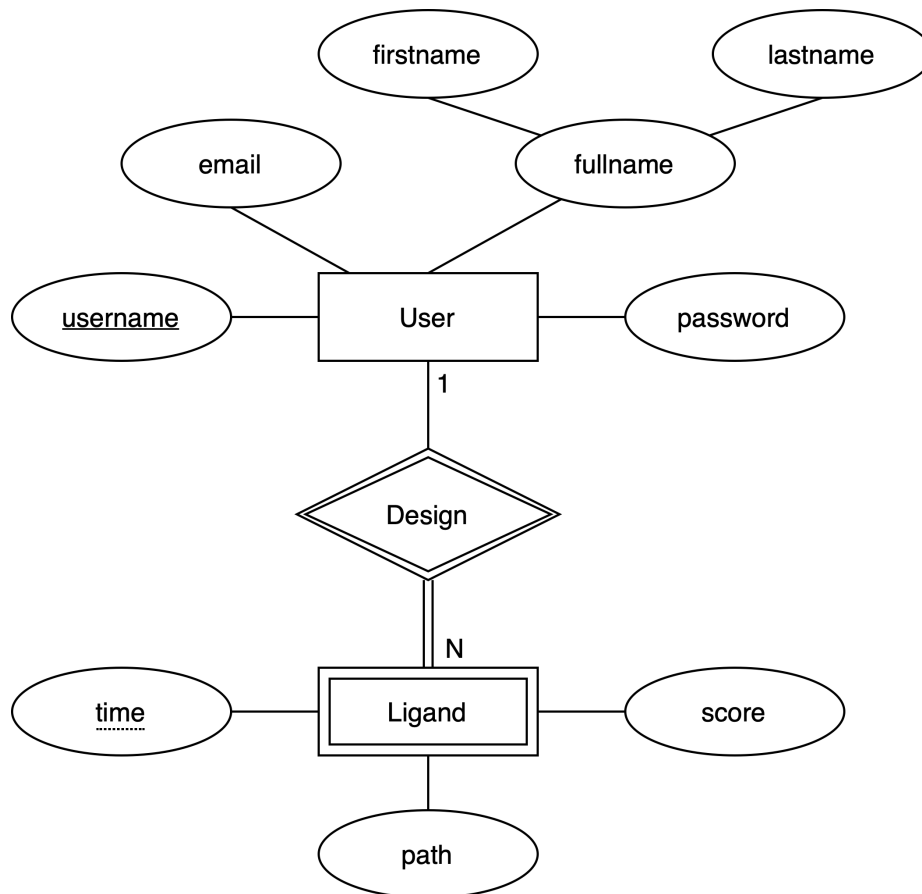


Figure 4.12: EERD of database

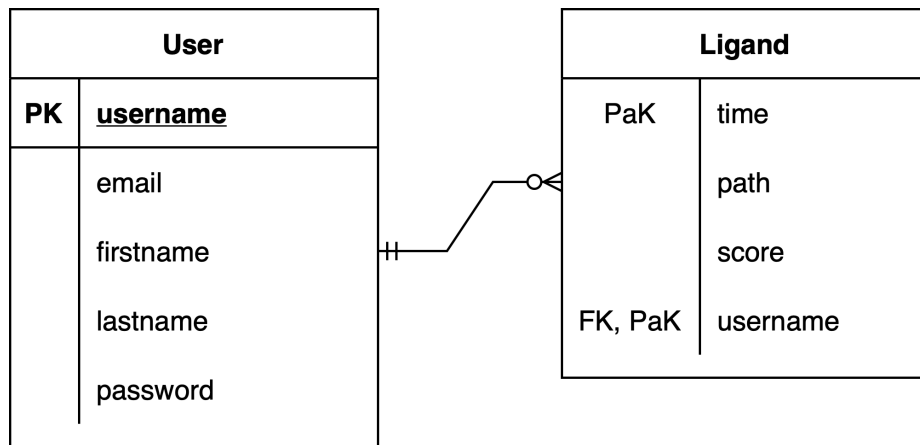


Figure 4.13: Database design schema

# Chapter 5

## Experiments and Results

### 5.1 Performance of The Propsoed GNN-based Model

#### 5.1.1 Pharmacologically Active Predicting

#### Datasets and Configurations

Currently, all available datasets that can be used to validate DTI methods partially overlap with others. The most reliable dataset is DrugBank [71] with 5260 drug-target complexes including 1317 confirmed active ones. But the main problem is that our chosen baseline method requires the co-crystal data (the 3D structure) when a protein interacts with a ligand while the DrugBank dataset lacks this information. So, we need to move to the others. The largest dataset is the ChEMBL [72]. It includes millions of ligands with verified activities but the same as the DrugBank one, it does not fully include the co-crystal data. The most favourite dataset is the Protein Data Bank [67] and its variants such as PDBbind [73], BindingDB [74], scPDB [75], etc. These datasets do not have only the co-crystal data as required by our baseline but also the verified binding affinity and protein pocket. There are many researchers that used these datasets to evaluate [9, 14, 64, 76]. According to the above analysis, we choose the PDBbind dataset, a variant of Protein Data Bank, as the data to pretrain the baseline method. Toward the data specified for Coronavirus, we use the data of Fragalysis which is the most largest open available experimented data upto now.

We summarize two datasets that are used to train and test our model in Table 5.1.

In a more details, PDBbind dataset is used for pretraining while Fragalysis dataset is used for finetuning our model. We also use the Fragalysis to directly train our model for demonstrating the affect of transfer learning. On one hand, the PDBbind dataset is splitted into training and testing sets with a ratio of 8:2. On the other hand, the Fragalysis dataset is splitted into training and testing sets with a ratio of 7:3.

Table 5.1 also indicates the number of active and inactive compounds in each dataset. A complex of protein and compound is labeled active or inactive based on its IC50. In case IC50 of a complex is equal to or lower than  $2.5\mu M$  it is considered as active and otherwise, it is inactive. Due to the disproportion in the number of active and inactive samples, we implement the undersampling technique to the class with higher quantity to swipe out the bias in training process.

**Table 5.1:** The number of protein-compound complexes used for training and testing of each dataset

	PDBbind			Fragalysis		
	Active	Inactive	Total	Active	Inactive	Total
Training	10037	5237	15274	75	125	200
Testing	2530	1287	3817	35	54	89

Before showing the training results, we want to specify the hyperparameters that are used in model implementation. The models are composed of 4 GNN layers, the dimension of each node embedding vector in GNN layer is 140. Our implementation uses the same number of fully connected layers in the final classifier as the original study which is 4 and the dimension of those layers equals 128 except the final one is only one (due to the binary classification problem) [14]. We use a batch size of 16 in the training process. To avoid overfitting, we also adopt the dropout rate to 0.3. Regarding the training of models that applys Improvement 3, the number of hops is set to 5, which is proven to produce better results according to [69]. The final hyperparameter is the number of epochs, which differ while training on different datasets and are not dependent on the models. When training directly on Fragalysis, we choose the checkpoint with the highest AUC score out of the first 1000 epochs. As for the pretrain model, the best checkpoint among the first 50 epochs will be selected. Then, that checkpoint will be used to finetune on Fragalysis and the best result of the first 1000 epochs will be chosen. All of our experments are performed on Google

Colab with 2-core CPU, Tesla K80 GPU and 12GB of RAM.

To objectively validate our improved model performance, we perform comparisons with other deep learning-based models. Particularly, we choose top-tier models which requires different input representations including DeepDTA (string-based) [63], DrugVQA (string-based for compound + feature matrix for protein) [36] and GraphDTA (graph-based for compound + string-based for protein) [61]. These methods were reproduced using official implementations from authors and the best results are chosen to report. We use the metric of *Area Under the ROC Curve* (AUC score) [77] to judge overall performance between models. Table 5.2 summarizes the AUC score of the above models and our model in different settings and scenarios. The first column indicates the models with their settings. The last three columns store the AUC scores of models when being directly trained on the Fragalysis, pretrained on PDBbind and finetuned on Fragalysis dataset, respectively.

**Table 5.2:** AUC scores of baseline model compared to other models in various settings grouped by molecular representation type

Model with settings	Directly trained on Fragalysis	Pretrained on PDBbind	Finetuned on Fragalysis
<i>String-based representation</i>			
DeepDTA	0.870	<b>0.849</b>	0.862
<i>String-based + Feature matrix representation</i>			
DrugVQA	0.853	0.819	0.820
<i>Graph-based + String-based representation</i>			
GraphDTA-GINConvNet	0.885	0.838	0.874
GraphDTA-GATNet	<b>0.886</b>	0.814	0.890
GraphDTA-GCNet	0.868	0.836	0.862
GraphDTA-GAT_GCN	0.874	0.835	0.874
<i>Graph-based representation</i>			
Baseline model	0.841	0.758	0.859
Baseline + Ipmt 1	0.865	0.787	0.896
Baseline + Ipmt 2	0.877	0.785	0.915
Baseline + Ipmt 3	0.870	0.793	0.936
Baseline + Ipmt 1,2	0.822	0.813	0.930
Baseline + Ipmt 1,2,3	0.868	0.820	<b>0.938</b>



## The Benefit of Transfer Learning Strategy

In Table 5.2, we can observe improvements in accuracy when our models have been pre-trained on the PDBbind compared to directly trained on the Fragalys. Specifically, toward the baseline model, the AUC score increases from 0.841 to 0.859 (increased 0.018). With settings including our improvements, the AUC scores make a big leap, where, with the two first improvements, the difference is significant up to 0.108. This suggests that without the pretraining process, the models generally can't learn how compounds interact with protein and seem overfitted when our improvements are applied. This is clearly seen by the combination of Improvement 1 and 2 only achieves an AUC score of 0.822 which is lower than that of the baseline model (0.841), while in the finetuning scenario, it achieves better performance than the baseline one ( $0.930 > 0.859$ ). Although the results when being pretrained on the PDBbind dataset are not too high, the pretraining process helps our models increase their generalizability and learn the general interaction principles. Therefore, the finetuning results on the Fragalys are not overfitted and achieve better performance than results of directly trained models.

## The Effects of Proposed Improvements

According to Table 5.2, when applying each improvement separately, the produced results are better than that of the baseline model in all scenarios. In a more insightful view, when applying each improvement separately, the results always have higher AUC scores than that of the baseline model. The most effective improvement belongs to the multi-hop gating mechanism. With this mechanism, finetuned model achieves 0.936 (improved 0.077 from the baseline). Moreover, when our improvements are combined together, they can boost the model performance to a higher level. The combination of Improvement 1 and 2 results in AUC scores of 0.813 and 0.930 which are better than those of single improvement configurations in PDBbind pretraining scenario and in Fragalys finetuning scenario respectively. Toward the best results our models achieved, the setting including all three improvements reaches an outstanding score of 0.938 in comparison with the scores from other settings in the finetuning scenario. From the above observations, we can consider that the multi-hop gating mechanism is really effective in creating a larger influential field for an atom or group of atoms. Thus, this helps our models better generalize the functional groups in both

protein and compound, which leads to outstanding results.

## Comparing to Other Methods

When being trained directly on Fragalysis, the combination of the baseline model and Improvement 2 achieves 0.877 AUC score, which is better than that of DeepDTA and DrugVQA, at 0.870 and 0.853, respectively. However, the GraphDTA method has a setting that reaches 0.886, which is higher than all of our models. In case being pretrained on PDBbind, our models show slightly lower performance. Specifically, the highest AUC score of our models is 0.820, which is lower than almost AUC scores of GraphDTA and lower than that of DeepDTA (0.849). In spite of the unsatisfied results on the PDBbind dataset, our models outperform the others when being finetuned on the Fragalysis dataset. In detail, all settings that have our improvements achieve AUC scores from 0.896 up to 0.938, which are strictly higher than the highest AUC score of GraphDTA (0.890), DeepDTA (0.862), and DrugVQA (0.820). These outstanding results suggest that our improved models are good at learning the general interaction principles and thus, when being finetuned on Fragalysis, they achieve better results than other methods.

### 5.1.2 Drug-Protein Pairwise Interactions Prediction

According to our analysis in the above chapter, currently, no dataset annotates detailed pairwise interactions between protein and ligand for us to directly validate our method. Therefore, we incorporated a research group from the University of Medicine and Pharmacy at Ho Chi Minh City to create a test set containing 20 ligands with annotated common pairwise interactions to target Coronavirus protein. Each groundtruth contains multiple pairs of ligand atom and protein atom that are verified by experts in pharmacy to have interaction. We compare our predicted interactions on each ligand and get the average result for the whole test set. The performance of our method to identify pairwise interactions is shown in Table 5.3 in the metric of *Recall*. Because of the complicated chemical interactions, validating our method with the *Precision* metric is not fair because there will be some cases that our model identified an actual interaction while it does not easy to discover and does not appear in the ground truth of the test set. Another reason we do not validate the *Precision* is due to the final goal of this thesis. We want to explore as many interactions as possible to create a thoughtful cavity model for the target protein.

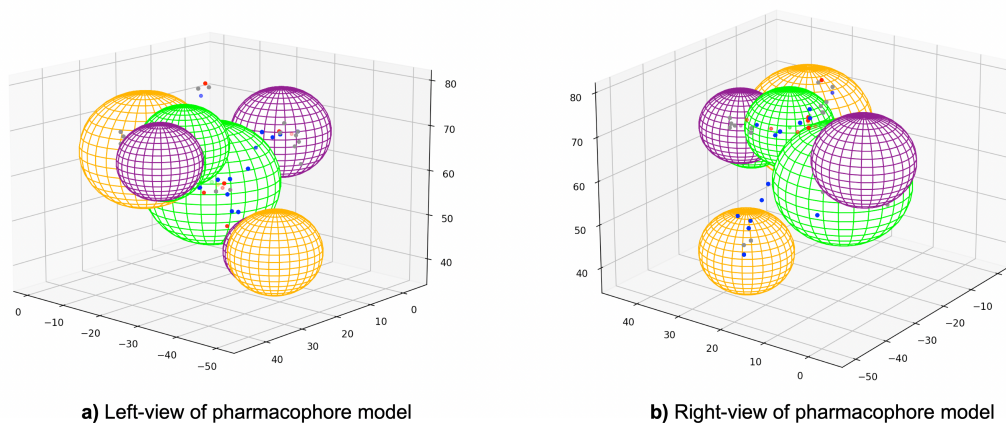
**Table 5.3:** The evaluation results of pairwise interaction prediction

No.	Complex name	Recall	No.	Complex name	Recall
1	6LZE	0.82	11	6XQS	0.93
2	6M0K	0.58	12	7E19	0.73
3	6WTK	1.00	13	7JU7	0.21
4	6XA4	0.56	14	7KX5	0.86
5	6XBG	1.00	15	7L0D	0.86
6	6XBH	1.00	16	7LMD	0.50
7	6XBI	0.94	17	7LME	0.63
8	6XCH	1.00	18	7LMF	0.83
9	6XFN	0.40	19	7LMH	1.00
10	6XHM	0.88	20	7LMJ	1.00
<b>Average Recall</b>				<b>0.79</b>	

## 5.2 The Cavity Model of Target Protein

Using the dataset from ViDok , we have created the pharmacophore model (cavity model in general) for the target protein of COVID-19 using our DBSCAN-based algorithm. Firstly, we crawl top 1000 complexes (including protein and ligand) that have highest docking scores. Then, we apply our proposed model in the first component to extract pairwise interactions. Figure 5.2 visualizes the protein atoms associated with interactions in 3D space. After that, we cluster the locations in 3D space of protein atoms having interactions to spheres to create pharmacophore model. Toward the modified DBSCAN algorithm, we set the hyperparameters of radius for neighbors  $\epsilon$  to  $5.5\text{\AA}$  and minimum number of cluster points  $minPts$  to 400.

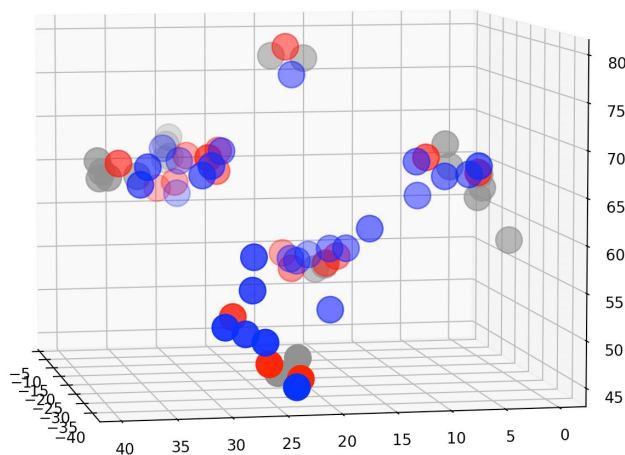
As presented in the above chapters, the pharmacophore model contains multiple spheres and each sphere represents for a group of atoms that have the same feature. Our resulted list of spheres is shown in Figure 5.4. In this table, each sphere is presented in each row with its coordinates and feature. Figure 5.1 visualizes all the spheres of our pharmacophore model in 3D space. In Figure 5.1, spheres with *Hydrogen Donor* feature are colored as green while spheres with *Hydrogen Acceptor* feature are colored as orange. The color purple is used to indicate the spheres with *Hydrophobic* feature.



**Figure 5.1:** Our built pharmacophore model

**Table 5.4:** Built pharmacophore model of the target protein

Feature	X	Y	Z	Radius
Hydrogen Donor	-30.225	3.831	66.757	5.5
Hydrogen Donor	-19.0313	16.6303	56.0029	13.623
Hydrogen Donor	-21.6388	27.2708	66.9083	8.8003
Hydrogen Donor	-24.0361	34.8547	64.6313	7.628
Hydrogen Donor	-11.6326	29.4802	66.0156	7.2845
Hydrogen Donor	-8.198	28.631	61.431	5.5
Hydrogen Acceptor	-41.5963	23.2133	46.3095	9.1333
Hydrogen Acceptor	-20.4093	16.358	55.6273	11.5058
Hydrogen Acceptor	-12.8375	28.9957	64.8636	12.7014
Hydrophobic	-40.0793	24.85	46.7966	7.7066
Hydrophobic	-27.6896	5.9601	66.6768	9.9262
Hydrophobic	-24.5747	36.9257	65.9457	8.211
Hydrophobic	-20.9697	17.4683	54.4411	9.3026
Hydrophobic	-7.5645	29.226	65.6474	7.8231



**Figure 5.2:** Our predicted interacting protein atoms using top 1000 complexes from ViDok. The blue, red and gray colors indicate *Hydrogen Donor*, *Hydrogen Acceptor* and *Hydrophobic* interaction, respectively.

### 5.3 The Performance of Decision Support System

In order to verify the effectiveness of our decision support system, we perform analysis on real persons who are not pharmacologists and have basic chemistry knowledge. There were 26 persons taken in our experiment. Each person is required to use their chemical knowledge or a random amino acid as an initial template to design ligands. After they submit their designs, our system will perform docking and generate suggestions for them. They will receive the docking score, the 3D structure of the ligand-protein complex, and our suggestions. They are also required to apply a random suggestion in the returned list and resubmit their design. In this way, we can evaluate how good our suggestions are or how our system helps boost the docking score. We have summarized the information about participants in our experiment in Table 5.5.

**Table 5.5:** Information about participants in our experiment

Number of participants	26
Total designed ligands	200
Number of ligand per participants	2-22
Average number of ligands per participants	7.69

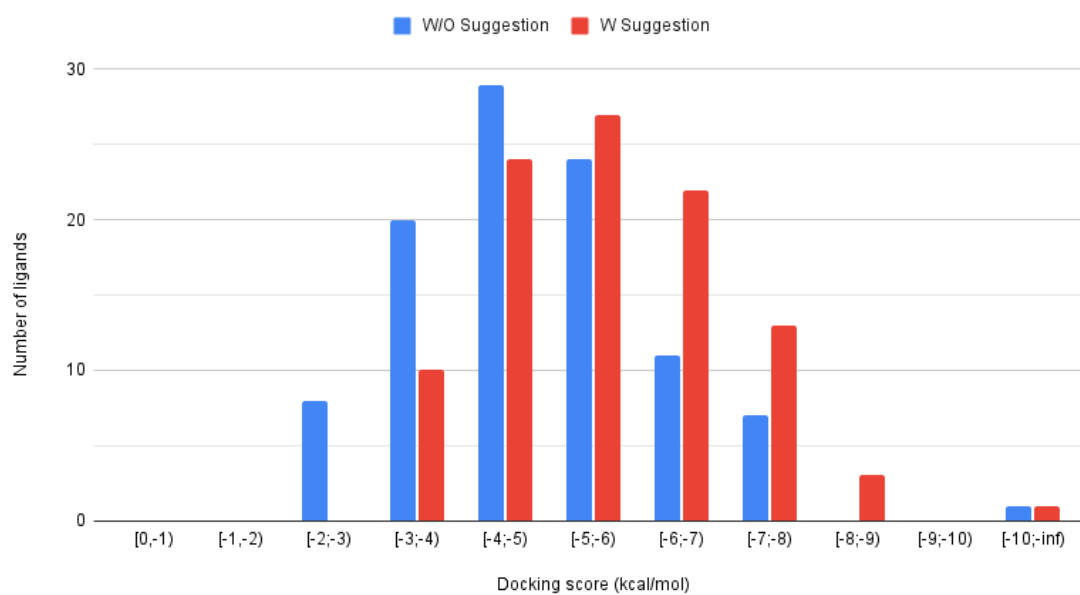
Because we strictly require participants to apply a suggestion for their design, the number of design ligands with and without suggestion are the same and equal to 100. By comparing the docking between each pair of ligands (with and without suggestion), we obtain how much the docking score is improved by our suggestion for each pair. Then, we calculate the average score for all 100 ligands in each group. The results of this evaluation are presented in Table 5.6. We also visualized the histogram of the docking scores of ligands with and without suggestion in Figure 5.3. Please notice that the docking score reflects the binding affinity of the ligand to the target protein, so that the lower the docking score is the better the ligand is.

According to Table 5.6 and Figure 5.3, we can consider that our suggestions can help boosting the docking score in an average of  $-0,762$  kcal/mol. The average scores after applying suggestions are  $-5.672$ , which is 16.6% better than the initial design from the participants. These results are strong evidence proving our proposed pipeline.

**Table 5.6:** Experiment results about the performance of our decision support system

	W/O Suggestion	W Suggestion
Number of ligands	100	100
Mean docking score	-4.91	-5.67
Standard deviation of docking scores	1.46	1.41
Average improved score	-0.76	
Average improved percentage	16.6%	

Histogram of docking score of designed ligand with and without suggestion

**Figure 5.3:** Histogram of docking scores between ligands with and without suggestion

## 5.4 Web Application

### 5.4.1 Screenshots on Multiple Platforms

Our developed web application has four important areas as i) The area for ligand designing tools; ii) The area showing suggestions; iii) The area showing 3D structure and docking score and iv) The scoreboard containing designed ligands from all users. Figure 5.4, 5.5, 5.6 and 5.7 are the screenshots of these areas respectively. We also performed testing our application on multiple platforms. The screenshots of our application on many different platforms with different browsers are shown in Figure 5.8.



Figure 5.4: Screenshot of the ligand designing tool

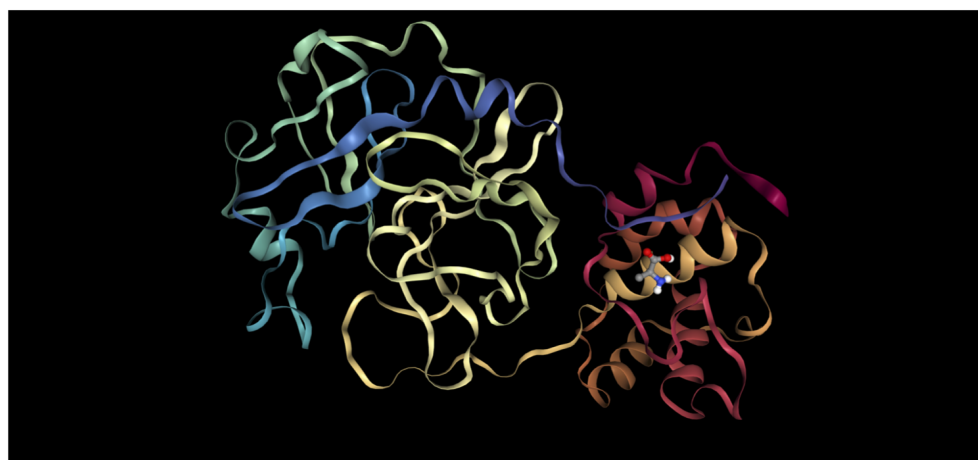


Suggestions ×

Reset

No.	Type	Atom index(es)	New Atom(s)	Apply
0	replace	14	O=CO	<span style="border: 1px solid blue; border-radius: 10px; padding: 2px 5px;">Apply</span>
1	replace	14	C=O	<span style="border: 1px solid blue; border-radius: 10px; padding: 2px 5px;">Apply</span>
2	replace	14	NC=O	<span style="border: 1px solid blue; border-radius: 10px; padding: 2px 5px;">Apply</span>
3	add	13	O=CO	<span style="border: 1px solid blue; border-radius: 10px; padding: 2px 5px;">Apply</span>
4	add	13	C=O	<span style="border: 1px solid blue; border-radius: 10px; padding: 2px 5px;">Apply</span>
5	add	13	NC=O	<span style="border: 1px solid blue; border-radius: 10px; padding: 2px 5px;">Apply</span>
6	add	4	O=CO	<span style="border: 1px solid blue; border-radius: 10px; padding: 2px 5px;">Apply</span>
7	add	4	C=O	<span style="border: 1px solid blue; border-radius: 10px; padding: 2px 5px;">Apply</span>

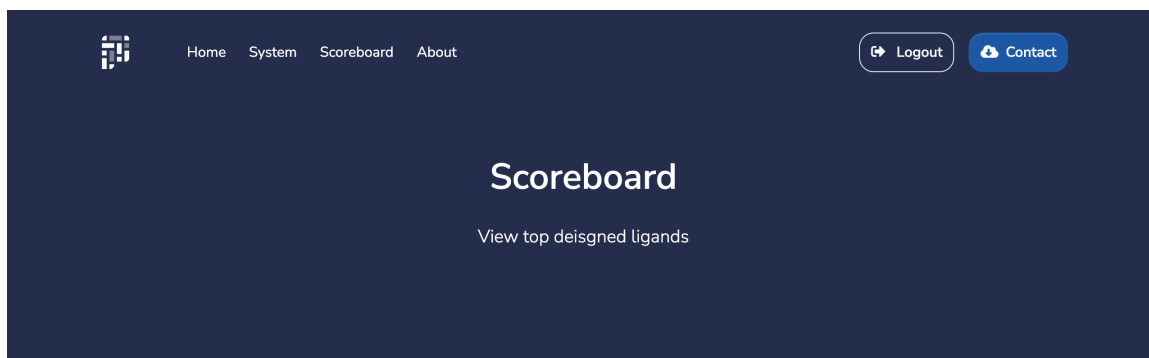
**Figure 5.5:** Screenshot of the suggestion area



Docking Score: -3.388

Runtime: 9.583s

**Figure 5.6:** Screenshot of the result area



Designed Ligands Ligands per page

No.	Name	Score	User	Download
1	DOCKED2022-05-09T08:43:29Z.mol	-11.92	cuong	<input type="button" value="Download"/>
2	DOCKED2022-05-09T08:42:37Z.mol	-11.402	cuong	<input type="button" value="Download"/>
3	DOCKED2022-05-11T20:27:29Z.mol	-11.147	Duy Phuoc	<input type="button" value="Download"/>
4	DOCKED2022-05-11T20:26:44Z.mol	-10.841	Duy Phuoc	<input type="button" value="Download"/>
5	DOCKED2022-05-11T20:25:06Z.mol	-10.818	Duy Phuoc	<input type="button" value="Download"/>
6	DOCKED2022-05-11T20:26:14Z.mol	-10.601	Duy Phuoc	<input type="button" value="Download"/>
7	DOCKED2022-05-11T20:15:25Z.mol	-10.352	Duy Phuoc	<input type="button" value="Download"/>
8	DOCKED2022-05-11T20:23:51Z.mol	-10.281	Duy Phuoc	<input type="button" value="Download"/>
9	DOCKED2022-05-11T20:14:43Z.mol	-10.272	Duy Phuoc	<input type="button" value="Download"/>
10	DOCKED2022-05-11T20:11:52Z.mol	-10.265	Duy Phuoc	<input type="button" value="Download"/>

Figure 5.7: Screenshot of the scoreboard

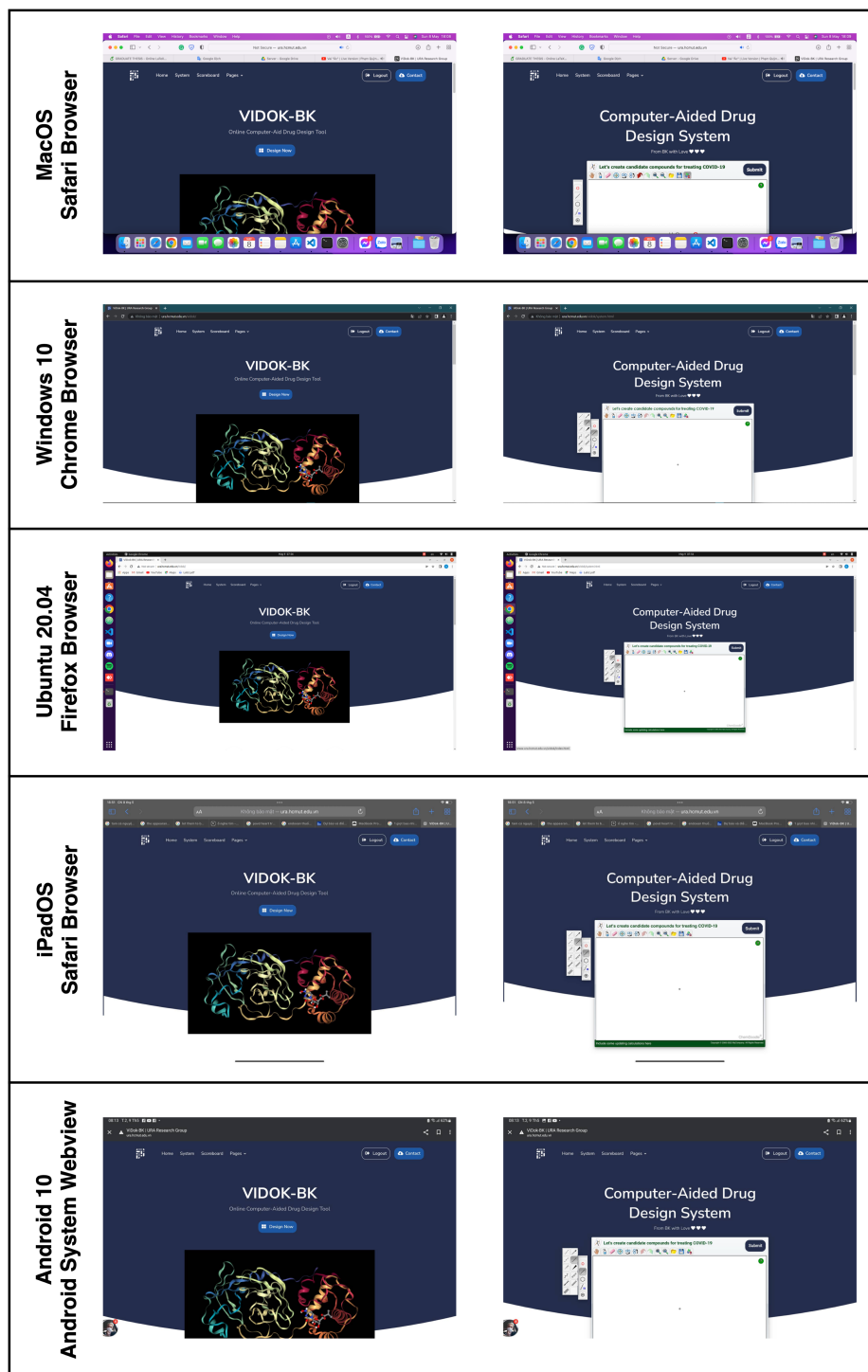


Figure 5.8: Screenshots of our web application on multiple platforms

### 5.4.2 Computing Time

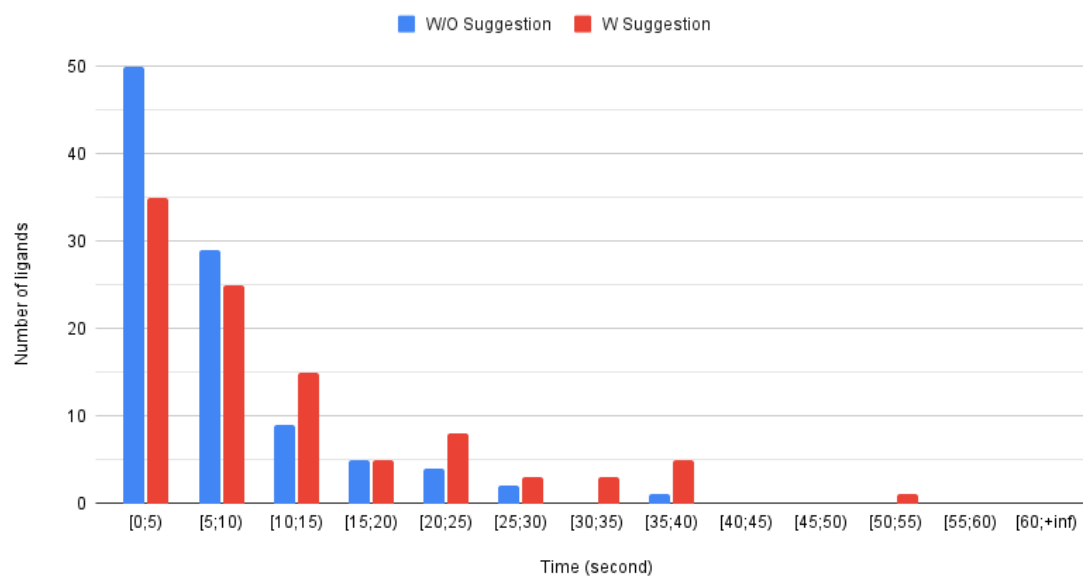
One of the most important factors that affects the drug discovery process is time. Therefore, to prove the efficiency of our system, we perform time-testing experiments. In this experiment, we reuse that data from 26 participants and 200 designed ligands as in the above section. When the participants submit their design, we trace the time our system performs docking calculation and suggestion generating. The detailed results are shown in Table 5.7. Moreover, we also compare the time for processing ligand with and without suggestion. Figure 5.9 visualizes the histogram for this comparison.

**Table 5.7:** The results of our time-tesing experiment

	W/O Suggestion	W Suggestion
Number of ligands	100	100
Mean processing time	7.07	11.64
Standard deviation of processing time	6.56	10.98
Average additional time	4.57	
Average additional percentage	66.5%	

As in the above results, we can consider that our developed system satisfies the predefined non-functional requirement about processing time (5 minutes). Besides, according to Figure 5.9, the time for processing ligands with suggestion is always higher than the time for ones without suggestion. It is a foreseen behavior because applying suggestion usually makes the ligands more complex, which leads to a longer processing time.

Histogram of processing time for scoring designed ligands with and without suggestion

**Figure 5.9:** Histogram of processing time for ligands with and without suggestion

# Chapter 6

## Conclusion

### 6.1 Summary

Technology in general and Computer Science in particular are developing rapidly, followed by the development of other industries. As a result, recent advances in all fields take the form of new techniques in computer science. Drug discovery, in general, is an arduous, complex process that requires meticulousness and creativity both in research and testing. These particular characteristics make the drug design process lengthy and costly. Since then, crowdsourcing drug design had become popular in the past 2-3 years, when the COVID-19 epidemic was raging worldwide. To make the most of the community's creativity in crowdsourcing drug design, a system of instructions and suggestions for designers who do not have much chemical knowledge is an option worth considering. In this thesis, I have used techniques based on Artificial Intelligence to build a highly accurate predictive model of drug-protein interactions. From these predicted interactions, I have successfully created a specific pharmacophore model representing the Coronavirus. The algorithm that gives suggestions for drug design, developed by me based on the pharmacophore model and chemical knowledge, has increased the binding affinity of the designed drug, making the drug better able to bind to the target protein of Coronavirus. Our system is not expected to be used by only pharmacologists but also by everyone having basic chemical knowledge. In addition, our website for drug design support system is deployed at <http://www.ura.hcmut.edu.vn/vidok> so that everyone can easily use it.

## 6.2 Future Developments

In the history of human development, there have been many diseases and epidemics occurring, such as the SAR pandemic (2002), African swine fever, HIV/AIDS, and most recently, the COVID-19 pandemic. These epidemics have serious consequences, and history has shown us that finding an excellent drug to fight the viruses that cause these diseases takes a lot of time and effort. For example, toward the disease of the century HIV/AIDS, until now, there is no specific drug that can completely destroy the virus that causes this disease. Therefore, the potential of drug design systems is immense and can be applied to many different viruses. Our drug design decision support system is similarly capable. With the techniques we have successfully applied to Coronavirus data, we believe that the system can be wholly applied to many other viruses to help the process of finding new drugs quickly and economically. Moreover, our system can apply more advanced Artificial Intelligence techniques to achieve better recommendation performance. In particular, improvements can be applied, such as improving the drug-protein interaction prediction model to analyze more types of interactions, improving the algorithm for generating suggestions so that we can rank which suggestions are good, and building a generative model based on AI when having enough data to create better drugs directly.



# References

- [1] X. Qing, X. Lee, J. De-Raeymaecker, J. Tame, K. Zhang, M. De-Maeyer, and A. Voet, “Pharmacophore modeling: advances, limitations, and current utility in drug discovery,” *Journal of Receptor, Ligand and Channel Research*, vol. 7, pp. 81–92, 2014.
- [2] G. Leroux-Roels, P. Bonanni, T. Tantawichien, and F. Zepp, “Vaccine development,” *Perspectives in Vaccinology*, vol. 1, no. 1, pp. 115–150, 2011.
- [3] J. Drews and S. Ryser, “Drug Development: The role of innovation in drug development,” *Nature Biotechnology*, vol. 15, pp. 1318–1319, Dec. 1997.
- [4] The Univeristy of Utah, “Vidok.” <https://vidok.chpc.utah.edu>, 2021.
- [5] D. K. Johnson and J. Karanicolas, “Druggable Protein Interaction Sites Are More Predisposed to Surface Pocket Formation than the Rest of the Protein Surface,” *PLOS Computational Biology*, vol. 9, pp. 1–10, 03 2013.
- [6] M. Aleksander and J. Renata, *Decision Support Systems*, ch. Decision Support Systems for Pharmaceutical Formulation Development Based on Artificial Neural Networks. IntechOpen, Jan. 2010.
- [7] O. Aytun and O. Melih, “A Drug Decision Support System for Developing a Successful Drug Candidate Using Machine Learning Techniques,” *Current Computer-Aided Drug Design*, vol. 16, no. 4, 2020.
- [8] Z. Deng, C. Chuaqui, and J. Singh, “Structural Interaction Fingerprint (SIFt): A Novel Method for Analyzing Three-Dimensional Protein-Ligand Binding Interactions,” *Journal of Medicinal Chemistry*, vol. 47, no. 2, pp. 337–344, 2004.

- [9] M. Thafar, A. B. Raies, S. Albaradei, M. Essack, and V. B. Bajic, "Comparison Study of Computational Prediction Tools for Drug-Target Binding Affinities," *Frontiers in Chemistry*, vol. 7, 2019.
- [10] K. E. Riley and P. Hobza, "Noncovalent interactions in biochemistry," *WIREs Computational Molecular Science*, vol. 1, no. 1, pp. 3–17, 2011.
- [11] S. Raghunathan and U. D. Priyakumar, "Molecular representations for machine learning applications in chemistry," *International Journal of Quantum Chemistry*, vol. 122, no. 7, p. e26870, 2022.
- [12] S. Lim, Y. Lu, C. Y. Cho, I. Sung, J. Kim, Y. Kim, S. Park, and S. Kim, "A review on compound-protein interaction prediction methods: Data, format, representation and model," *Computational and Structural Biotechnology Journal*, vol. 19, pp. 1541–1556, 2021.
- [13] I. Muegge and P. Mukherjee, "An overview of molecular fingerprint similarity search in virtual screening," *Expert Opinion on Drug Discovery*, vol. 11, no. 2, pp. 137–148, 2016.
- [14] J. Lim, S. Ryu, K. Park, Y. J. Choe, J. Ham, and W. Y. Kim, "Predicting Drug-Target Interaction Using a Novel Graph Neural Network with 3D Structure-Embedded Graph Representation," *Journal of Chemical Information and Modeling*, vol. 59, no. 9, pp. 3981–3988, 2019.
- [15] F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner, and G. Monfardini, "The Graph Neural Network Model," *IEEE Transactions on Neural Networks*, vol. 20, no. 1, pp. 61–80, 2009.
- [16] P. Velickovic, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, "Graph attention networks," in *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*, OpenReview.net, 2018.
- [17] L. Brigato and L. Iocchi, "A Close Look at Deep Learning with Small Data," in *2020 25th International Conference on Pattern Recognition (ICPR)*, pp. 2490–2497, 2021.

- [18] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*, ch. Representation Learning, p. 536. Cambridge, MA, US: MIT Press, 2016.
- [19] J. L. Bentley, “A Survey of Techniques for Fixed Radius near Neighbor Searching,” tech. rep., Stanford University, Stanford, CA, USA, 1975.
- [20] J. L. Bentley, D. F. Stanat, and E. Williams, “The complexity of finding fixed-radius near neighbors,” *Information Processing Letters*, vol. 6, no. 6, pp. 209–212, 1977.
- [21] S. M. Omohundro, “Five Balltree Construction Algorithms,” tech. rep., University of California, Berkeley, Berkeley, CA, USA, 1989.
- [22] J. L. Bentley, “Multidimensional Binary Search Trees Used for Associative Searching,” *Commun. ACM*, vol. 18, p. 509–517, sep 1975.
- [23] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, “A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise,” in *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, KDD’96, p. 226–231, AAAI Press, 1996.
- [24] P. Schneider and G. Schneider, “De novo design at the edge of chaos,” *Journal of Medicinal Chemistry*, vol. 59, no. 9, pp. 4077–4086, 2016.
- [25] R. V. Devi, S. S. Sathya, and M. S. Coumar, “Evolutionary algorithms for de novo drug design – a survey,” *Applied Soft Computing*, vol. 27, pp. 543–552, 2015.
- [26] A. C. Nicolaou, C. Kannas, and E. Loizidou, “Multi-objective optimization methods in de novo drug design,” *Mini-Reviews in Medicinal Chemistry*, vol. 12, no. 10, pp. 979–987, 2012.
- [27] C. A. Nicolaou and N. Brown, “Multi-objective optimization methods in drug design,” *Drug Discovery Today: Technologies*, vol. 10, no. 3, pp. e427–e435, 2013.
- [28] A. Sánchez-Rodríguez, Y. Pérez-Castillo, S. C. Schürer, O. Nicolotti, G. F. Mangiatordi, F. Borges, M. N. D. Cordeiro, E. Tejera, J. L. Medina-Franco, and

- M. Cruz-Monteagudo, “From flamingo dance to (desirable) drug discovery: a nature-inspired approach,” *Drug Discovery Today*, vol. 22, no. 10, pp. 1489–1502, 2017.
- [29] V. D. Mouchlis, A. Afantitis, A. Serra, M. Fratello, A. G. Papadiamantis, V. Aidinis, I. Lynch, D. Greco, and G. Melagraki, “Advances in De Novo Drug Design: From Conventional to Machine Learning Methods,” *International Journal of Molecular Sciences*, vol. 22, no. 4, 2021.
- [30] D. J. Danziger, P. M. Dean, and A. W. Cuthbert, “Automated site-directed drug design: a general algorithm for knowledge acquisition about hydrogen-bonding regions at protein surfaces,” *Proceedings of the Royal Society of London. B. Biological Sciences*, vol. 236, no. 1283, pp. 101–113, 1989.
- [31] H.-J. Böhm, “LUDI: rule-based automatic design of new substituents for enzyme inhibitor leads,” *Journal of Computer-Aided Molecular Design*, vol. 6, pp. 593–606, Dec. 1992.
- [32] B. Waszkowycz, D. E. Clark, D. Frenkel, J. Li, C. W. Murray, B. Robson, and D. R. Westhead, “Pro.ligand: An approach to de novo molecular design. 2. design of novel molecules from molecular field analysis (mfa) models and pharmacophores,” *Journal of Medicinal Chemistry*, vol. 37, no. 23, pp. 3994–4002, 1994.
- [33] V. J. Gillet, G. Myatt, Z. Zsoldos, and A. P. Johnson, “SPROUT, HIPPO and CAESA: Tools for de novo structure generation and estimation of synthetic accessibility,” *Perspectives in Drug Discovery and Design*, vol. 3, pp. 34–50, Dec. 1995.
- [34] O. Vincent, A. Gutierrez-Nogués, A. Trejo-Herrero, and M.-A. Navas, “A novel reverse two-hybrid method for the identification of missense mutations that disrupt protein–protein binding,” *Scientific Reports*, vol. 10, Dec. 2020.

- [35] M. M. Sharifabad, R. Sheikhpour, and S. Gharaghani, “BRNS + SSFSM-DTI: A hybrid method for drug-target interaction prediction based on balanced reliable negative samples and semi-supervised feature selection,” *Chemometrics and Intelligent Laboratory Systems*, vol. 220, p. 104462, 2022.
- [36] Y. Zheng, Shuangjia and Li, S. Chen, J. Xu, and Y. Yang, “Predicting drug-protein interaction using quasi-visual question answering system,” *Nature Machine Intelligence*, vol. 2, pp. 134–140, Feb. 2020.
- [37] J. Jiménez, S. Doerr, G. Martínez-Rosell, A. S. Rose, and G. De Fabritiis, “DeepSite: protein-binding site predictor using 3D-convolutional neural networks,” *Bioinformatics*, vol. 33, pp. 3036–3042, May 2017.
- [38] Z. Luo, R. Wang, and L. Lai, “RASSE: A New Method for Structure-Based Drug Design,” *Journal of Chemical Information and Computer Sciences*, vol. 36, no. 6, pp. 1187–1194, 1996.
- [39] D. A. Pearlman and M. A. Murcko, “Concerts: Dynamic connection of fragments as an approach to de novo ligand design,” *Journal of Medicinal Chemistry*, vol. 39, no. 8, pp. 1651–1663, 1996.
- [40] J. Zhu, H. Fan, H. Liu, and Y. Shi, “Structure-based ligand design for flexible proteins: Application of new F-DycoBlock,” *Journal of Computer-Aided Molecular Design*, vol. 15, pp. 979–996, Nov. 2001.
- [41] M. Hartenfeller, H. Zettl, M. Walter, M. Rupp, F. Reisen, E. Proschak, S. Weggen, H. Stark, and G. Schneider, “Dogs: Reaction-driven de novo design of bioactive compounds,” *PLOS Computational Biology*, vol. 8, pp. 1–12, 02 2012.
- [42] T. Fujita, ed., *Rational approaches to computer drug design based on drug-receptor interactions*, vol. 23 of *Pharmacochemistry Library*, pp. 3–48. Elsevier, 1995.
- [43] S. Kwon, H. Bae, J. Jo, and S. Yoon, “Comprehensive ensemble in QSAR prediction for drug discovery,” *BMC Bioinformatics*, vol. 20, Oct. 2019.

- [44] C. Acharya, A. Coop, J. E. Polli, and A. D. MacKerell, “Recent advances in ligand-based drug design: Relevance and utility of the conformationally sampled pharmacophore approach,” *Current Computer-Aided Drug Design*, vol. 7, no. 1, pp. 10–22, 2011.
- [45] I. I. Baskin, “The power of deep learning to ligand-based novel drug discovery,” *Expert Opinion on Drug Discovery*, vol. 15, no. 7, pp. 755–764, 2020.
- [46] F. Palazzesi and A. Pozzan, *Deep Learning Applied to Ligand-Based De Novo Drug Design*, pp. 273–299. New York, NY: Springer US, 2022.
- [47] V. T. Sabe, T. Ntombela, L. A. Jhamba, G. E. Maguire, T. Govender, T. Naicker, and H. G. Kruger, “Current trends in computer aided drug design and a highlight of drugs discovered via computational techniques: A review,” *European Journal of Medicinal Chemistry*, vol. 224, p. 113705, 2021.
- [48] J. Chodera, A. A. Lee, N. London, and F. von Delft, “Crowdsourcing drug discovery for pandemics,” *Nature Chemistry*, vol. 12, pp. 581–581, Jul 2020.
- [49] BIOVIA, “Dassault Systèmes, Discovery Studio,” 2022.
- [50] O. Trott and A. J. Olson, “AutoDock Vina: Improving the speed and accuracy of docking with a new scoring function, efficient optimization, and multithreading,” *Journal of Computational Chemistry*, vol. 31, no. 2, pp. 455–461, 2010.
- [51] Schrödinger, LLC, “The PyMOL molecular graphics system, version 1.8,” November 2015.
- [52] Schrödinger, LLC, “Maestro.” <https://www.schrodinger.com/products/maestro>, 2022.
- [53] G. M. Morris, R. Huey, W. Lindstrom, M. F. Sanner, R. K. Belew, D. S. Goodsell, and A. J. Olson, “AutoDock4 and AutoDockTools4: Automated docking with selective receptor flexibility,” *Journal of Computational Chemistry*, vol. 30, no. 16, pp. 2785–2791, 2009.

- [54] R. A. Friesner, R. B. Murphy, M. P. Repasky, L. L. Frye, J. R. Greenwood, T. A. Halgren, P. C. Sanschagrin, and D. T. Mainz, “Extra Precision Glide: Docking and Scoring Incorporating a Model of Hydrophobic Enclosure for Protein-Ligand Complexes,” *Journal of Medicinal Chemistry*, vol. 49, no. 21, pp. 6177–6196, 2006. PMID: 17034125.
- [55] G. Jones, P. Willett, R. C. Glen, A. R. Leach, and R. Taylor, “Development and validation of a genetic algorithm for flexible docking,” *Journal of Molecular Biology*, vol. 267, no. 3, pp. 727–748, 1997.
- [56] G. Wolber and T. Langer, “Ligandscout: 3-d pharmacophores derived from protein-bound ligands and their use as virtual screening filters,” *Journal of Chemical Information and Modeling*, vol. 45, no. 1, pp. 160–169, 2005. PMID: 15667141.
- [57] Chemical Computing Group ULC, “Molecular Operating Environment (MOE).” <https://www.chemcomp.com/Products.htm>, 2022.
- [58] N. S. Pagadala, K. Syed, and J. Tuszynski, “Software for molecular docking: a review,” *Biophysical Reviews*, vol. 9, pp. 91–102, Apr 2017.
- [59] A. Dhakal, C. McKay, J. J. Tanner, and J. Cheng, “Artificial intelligence in the prediction of protein–ligand interactions: recent advances and future directions,” *Briefings in Bioinformatics*, vol. 23, 11 2021. bbab476.
- [60] A. Keshavarzi Arshadi, J. Webb, M. Salem, E. Cruz, S. Calad-Thomson, N. Ghadirian, J. Collins, E. Diez-Cecilia, B. Kelly, H. Goodarzi, and J. S. Yuan, “Artificial Intelligence for COVID-19 Drug Discovery and Vaccine Development,” *Frontiers in Artificial Intelligence*, vol. 3, 2020.
- [61] T. Nguyen, H. Le, T. P. Quinn, T. Nguyen, T. D. Le, and S. Venkatesh, “GraphDTA: predicting drug–target binding affinity with graph neural networks,” *Bioinformatics*, vol. 37, pp. 1140–1147, 10 2020.
- [62] I. Lee, J. Keum, and H. Nam, “DeepConv-DTI: Prediction of drug-target interactions via deep learning with convolution on protein sequences,” *PLOS Computational Biology*, vol. 15, pp. 1–21, 06 2019.

- [63] H. Öztürk, A. Özgür, and E. Ozkirimli, “DeepDTA: deep drug–target binding affinity prediction,” *Bioinformatics*, vol. 34, pp. i821–i829, 09 2018.
- [64] M. A. Thafar, R. S. Olayan, H. Ashoor, S. Albaradei, V. B. Bajic, X. Gao, T. Gojobori, and M. Essack, “DTiGEMS+: drug–target interaction prediction using graph embedding, graph mining, and similarity-based techniques,” *Journal of Cheminformatics*, vol. 12, p. 44, Jun 2020.
- [65] Z. Jin, X. Du, Y. Xu, Y. Deng, M. Liu, Y. Zhao, B. Zhang, X. Li, L. Zhang, C. Peng, Y. Duan, J. Yu, L. Wang, K. Yang, F. Liu, R. Jiang, X. Yang, T. You, X. Liu, X. Yang, F. Bai, H. Liu, X. Liu, L. W. Guddat, W. Xu, G. Xiao, C. Qin, Z. Shi, H. Jiang, Z. Rao, and H. Yang, “Structure of Mpro from SARS-CoV-2 and discovery of its inhibitors,” *Nature*, vol. 582, pp. 289–293, Jun 2020.
- [66] Diamond, “Fragalysis.” <https://fragalysis.diamond.ac.uk>, 2021. Accessed on Oct. 2021.
- [67] H. M. Berman, J. Westbrook, Z. Feng, G. Gilliland, T. N. Bhat, H. Weissig, I. N. Shindyalov, and P. E. Bourne, “The Protein Data Bank,” *Nucleic Acids Research*, vol. 28, pp. 235–242, 01 2000.
- [68] B. O’Neill, “Chapter 2 - frame fields,” in *Elementary Differential Geometry (Second Edition)* (B. O’Neill, ed.), pp. 43–99, Boston: Academic Press, second edition ed., 2006.
- [69] G. Wang, R. Ying, J. Huang, and J. Leskovec, “Multi-hop attention graph neural networks,” in *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21* (Z.-H. Zhou, ed.), pp. 3089–3096, International Joint Conferences on Artificial Intelligence Organization, 8 2021. Main Track.
- [70] G. Landrum, “RDKit: Open-Source Cheminformatics Software.” <https://www.rdkit.org>, 2010.
- [71] D. S. Wishart, Y. D. Feunang, A. C. Guo, E. J. Lo, A. Marcu, J. R. Grant, T. Sajed, D. Johnson, C. Li, Z. Sayeeda, N. Assempour, I. Iynkkaran, Y. Liu, A. Maciejewski, N. Gale, A. Wilson, L. Chin, R. Cummings, D. Le, A. Pon,



- C. Knox, and M. Wilson, “DrugBank 5.0: a major update to the DrugBank database for 2018,” *Nucleic Acids Research*, vol. 46, pp. D1074–D1082, 11 2017.
- [72] A. Gaulton, L. J. Bellis, A. P. Bento, J. Chambers, M. Davies, A. Hersey, Y. Light, S. McGlinchey, D. Michalovich, B. Al-Lazikani, and J. P. Overington, “ChEMBL: a large-scale bioactivity database for drug discovery,” *Nucleic Acids Research*, vol. 40, pp. D1100–D1107, 09 2011.
- [73] R. Wang, X. Fang, Y. Lu, C.-Y. Yang, and S. Wang, “The PDBbind Database: Methodologies and Updates,” *Journal of Medicinal Chemistry*, vol. 48, no. 12, pp. 4111–4119, 2005.
- [74] M. K. Gilson, T. Liu, M. Baitaluk, G. Nicola, L. Hwang, and J. Chong, “BindingDB in 2015: A public database for medicinal chemistry, computational chemistry and systems pharmacology,” *Nucleic Acids Research*, vol. 44, pp. D1045–D1053, 10 2015.
- [75] J. Desaphy, G. Bret, D. Rognan, and E. Kellenberger, “sc-PDB: a 3D-database of ligandable binding sites—10 years on,” *Nucleic Acids Research*, vol. 43, pp. D399–D404, 10 2014.
- [76] K. Y. Gao, A. Fokoue, H. Luo, A. Iyengar, S. Dey, and P. Zhang, “Interpretable drug target prediction using deep neural representation,” in *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18*, pp. 3371–3377, International Joint Conferences on Artificial Intelligence Organization, 7 2018.
- [77] A. P. Bradley, “The use of the area under the ROC curve in the evaluation of machine learning algorithms,” *Pattern Recognition*, vol. 30, no. 7, pp. 1145–1159, 1997.

# Appendix A

## Scientific paper

With the research presented in Section Identifying Drug-Protein Interactions, we have summarized the achieved results and written a scientific paper. The full text of the scientific paper is attached below.

Systems biology

# Towards *De Novo* Drug Design for the Coronavirus: A Drug-Target Interaction Prediction Approach using Atom-enhanced Graph Neural Network with Multi-hop Gating Mechanism

Duc Q. Nguyen<sup>1, 3</sup>, Khoan D. Le<sup>1, 3</sup>, Bach T. Ly<sup>1, 3</sup>, An D. Nguyen<sup>1, 3</sup>, Quang H. Nguyen<sup>1, 3</sup>, Tuan H. Nguyen<sup>2, 3</sup>, Cuong Quoc Duong<sup>4</sup>, Thanh N. Truong<sup>5</sup>,  
Phuong Thuy Viet Nguyen<sup>4</sup> and Tho T. Quan<sup>1, 3,\*</sup>

<sup>1</sup>Faculty of Computer Science and Engineering, Ho Chi Minh City University of Technology (HCMUT), Ho Chi Minh City, Vietnam,

<sup>2</sup>Faculty of Chemical Engineering, Ho Chi Minh City University of Technology (HCMUT), Ho Chi Minh City, Vietnam,

<sup>3</sup>Vietnam National University Ho Chi Minh City, Ho Chi Minh City, Vietnam,

<sup>4</sup>Faculty of Pharmacy, University of Medicine and Pharmacy at Ho Chi Minh City, Ho Chi Minh City, Vietnam, and

<sup>5</sup>University of Utah, Salt Lake City, Utah, United States.

\*Corresponding author: qttho@hcmut.edu.vn

Associate Editor: XXXXXXXX

Received on XXXXX; revised on XXXXX; accepted on XXXXX

## Abstract

COVID-19 pandemic, caused by Coronavirus, is undeniably a disaster for human beings. Although there are many effective vaccines developed, specific drugs are still in urgent demand. Normally, to identify new potential drugs, one needs to design compounds, and then test the interactions between them and the virus protein in an *in silico* manner for determining the drug candidates. This process, known as *Drug-Target Interaction* (DTI) prediction, can be used by molecular docking. However, the scoring function in molecular docking to capture interactions of compound-target interactions is still challenging. Therefore, it urges us to consider applying the latest *Artificial Intelligence* (AI) techniques for automating it. In particular, *Graph Neural Network* (GNN) attracts much attention since it is very suitable for the graph-based nature of compounds and virus proteins. However, to introduce a representation well-reflecting the biological structures of compounds and proteins for GNN is by no means a trivial task. Moreover, since the available datasets for Coronavirus are still limited, the recently developed GNN models have been suffering from overfitting when dealing with this kind of disease. In this paper, we address those issues by proposing a new model known as *Atom-enhanced Graph Neural Network with Multi-hop Gating Mechanism*. On one hand, our model captures more specifically biological information of compound and protein atoms. On the other hand, we introduce a new gating mechanism allowing the model to learn better from non-neighbor connections. Once applied with transfer learning from very large medical databanks, our model enjoyed outperformed performance experimenting with Coronavirus.

**Contact:** qttho@hcmut.edu.vn

**Keywords:** Coronavirus, Drug-Target Interaction, Graph Neural Networks, Multi-hop Gating Mechanism

## 1 Introduction

Viruses in general and *Coronavirus* (COVID-19) (WHO, 2021) in particular have caused many deaths in the world through recent years (Kontopantelis *et al.*, 2022). Besides vaccines, drugs are the keys for

humans to fight against pandemic waves caused by viruses. This makes drug design playing an extremely important role in constraining the pandemic, especially when many new variants may be possibly born (Burki, 2022).

To design a drug for a certain virus type, one may note that each type of virus has main specific proteins, which are medication for the assembly of replication-transcription machinery, in order to form new kernels for child viruses and could be an ideal antiviral target. Thus, drug design, in conceptualization, is a process of trial-and-error that creates many different *compounds* as illustrated in Figure 1. A compound, once designed, can be represented as a graph whose vertices and edges are *atoms* and *bonds* respectively, and so is the virus protein. The drug designer can try various designs by testing possible *interactions* between compound atoms with those from the virus protein. If a compound design introduces sufficiently many such key interactions with the virus protein, it is a high chance this compound may be effective to make the virus *inactivated*. Then, the drug designer can choose this compound design for the next step of drug production.

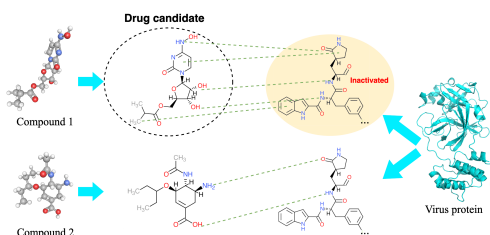


Fig. 1. The concept of interaction between drug and protein

To observe the interaction between a compound and a virus protein, traditional *in silico* screening methods usually rely on *docking tools* such as AutoDock Vina (Trott and Olson, 2010), Smina (Koes et al., 2013), MedusaDock 2.0 (Wang and Dokholyan, 2019), etc. However, as the docking process usually takes time (normally 3-5 minutes) and importantly limitation of scoring function, it is hard for a drug designer to explore a large scale of potential compounds to obtain the best one. With the recent advancements of *Artificial Intelligence* (AI) today, we can use AI models to predict the interactions between a compound design and a virus protein to select only highly-interactive compounds for *in vitro* experiments. It would significantly reduce time and cost for drug design process. This prediction task is often referred as *Drug-Target Interaction* DTI one.

Recent studies have demonstrated the strength of AI in drug design with various *Deep Learning* (DL) architectures to quickly and accurately remove known-weak-interaction compounds. As compounds and proteins are graph-based structures, graph analysis techniques are usually adopted in this area. In recent years, *Graph Neural Networks* (GNNs) (Scarselli et al., 2009) emerged as the latest advanced techniques for graph processing. Among state-of-the-art GNN-based models for DTI prediction, perhaps the study presented by Lim et al. (2019) is most remarkable as it does not only apply GNNs but also embeds the 3D structure of the compound and protein into graph features. This work will then serve as the baseline for our research.

Moreover, in the fight against COVID-19, there are not many compounds that have been put in *in vitro* screening. The largest crowd-sourcing drug discovery system for COVID-19, PostEra COVID Moonshot (Chodera et al., 2020), has experimented with only about 2000 compounds. In those experimented compounds, only 379 compounds have been

captured co-crystal data i.e. the 3D structure of compounds by Fragalysis (Diamond, 2021). As a consequence, current drug-for-COVID datasets lack generalization for training AI models, making them suffering from *overfitting*. We also address this issue in our study by proposing a transfer learning strategy in this research. Thus, our contributions are of two-fold as follows.

- Firstly, we apply transfer learning to pretrain the baseline model on a suitable known-interaction variant dataset of the large and diverse dataset from Protein Data Bank (Berman et al., 2000). Data from Protein Data Bank is collected from real experiments of pharmaceutical colleagues. Then we finetune the pretrained model for predicting interaction on COVID-19 target protein using the dataset of Fragalysis.
- Secondly, and importantly, we introduce a new model, known as *Atom-enhanced Graph Neural Network with Multi-hop Gating Mechanism*. This model is extended from a baseline model introduced in Lim et al. (2019), in which we introduced three major improvement strategies as follows: (i) *Enriched atom encoding*: we additionally encode atoms in the model with more important attributes; (ii) *Total atom aggregation*: we enhance the baseline model to aggregate not only the atoms of compounds but also of the proteins to produce more informative representation of the model output; and (iii) *Multi-hop gating mechanism*: we modify the original gating mechanism to allow non-neighbor atoms affect others, earning improved interaction prediction.

## 2 Preliminaries and Related Works

### 2.1 Drug-Target Interaction problem

The Drug-Target Interaction problem is a common problem in drug discovery and is solved by many methods from traditional to modern (Thafar et al., 2019). This problem takes a compound and a target protein as its inputs and uses an algorithm to predict if the compound can interact (be pharmacologically active) with the protein or not (binary classification) (Thafar et al., 2019). A compound, when being put together with a protein in an environment, can create bonds to that protein. A bond can belong to one of three major common types which are hydrogen, hydrophobic, Van Der Waals (Riley and Hobza, 2011). Each type has different characteristics and its own strengths. In case all bonds created have enough total strength (measured by a binding affinity metric like IC50), depending on the aims of the study, that compound can be considered as active or inactive based on a defined cut-off (Gao et al., 2018). For example, we want to keep only the compounds that have IC50 to the target protein less than  $1 \mu M$  for the *in vitro* screening, we can use  $1 \mu M$  as a threshold to split the datasets into active and inactive sets for the next analysis steps.

### 2.2 Modern DTI prediction techniques

Recent years show the vast developments of the AI Era in many distinct industries such as health, telecommunication, service industry, etc. Especially in the COVID-19 situation, many AI techniques from *Machine Learning* (ML) to *Deep Learning* (DL) such as Support Vector Machine, Ensemble Learning, CNN, Transformer, GNN, etc. were applied to help speed up the process of making vaccines and drugs (Keshavarzi Arshadi et al., 2020) and it makes no exception for the Drug-Target Interaction prediction problem (Lim et al., 2021). At the conceptualization level, almost recent DTI models initially use one suitable AI technique to encode the compound and protein into feature vectors based on their representations. Then, these models aggregate the two feature vectors together to feed into a classifier to predict if the compound is pharmacologically active with the protein or not (Lim et al., 2019;

Zheng *et al.*, 2020; Nguyen *et al.*, 2020; Lee *et al.*, 2019; Öztürk *et al.*, 2018). Some models also utilize the attention mechanism to enhance the performance when creating feature vectors (Lim *et al.*, 2019; Zheng *et al.*, 2020; Nguyen *et al.*, 2020).

Another approach for this DTI problem is based on the ‘‘guilt-by-association’’ principle (Thafar *et al.*, 2020; Muegge and Mukherjee, 2016). Methods belonging to this approach try to find similar proteins and compounds in the known-interaction datasets and then use them as evidence to predict the input compound-protein (Thafar *et al.*, 2020, 2019). This approach has the main weakness when dealing with strange proteins or compounds. In this case, there is not much ‘‘evidence’’ for guiding the model, so its performance could not be stable (Thafar *et al.*, 2019).

Comparing the above approaches, we can consider that with the unseen protein of Coronavirus, the second approach using similarities will not work as well as the first one with a proper training strategy. Therefore, we have decided to choose our baseline following the first approach (encoding the drug and protein than using the produced feature vector to classify).

### 2.3 Graph Attention Network Layer

A graph can be defined by  $(V, E)$ , where  $V$  is a set of nodes and  $E$  is a set of edges. The representation of a graph can be an adjacency matrix, adjacency list or incidence matrix. If the graph nodes have attributes, they will be represented by vectors. The *Graph Neural Network* (GNN), which was born in 2009 (Scarselli *et al.*, 2009), has been explored and developed in various different domains and has proved its noticeable performance in many applications. Many variants of GNNs are formed to adapt to specific domains. Usually, a GNN model contains multiple GNN layers. The input graph after being passed through  $N$  GNN layers will be aggregated to form a representation vector for the whole graph. Then, that vector can be used in various downstream tasks. A conceptual view of a GNN model is illustrated in Figure 2.

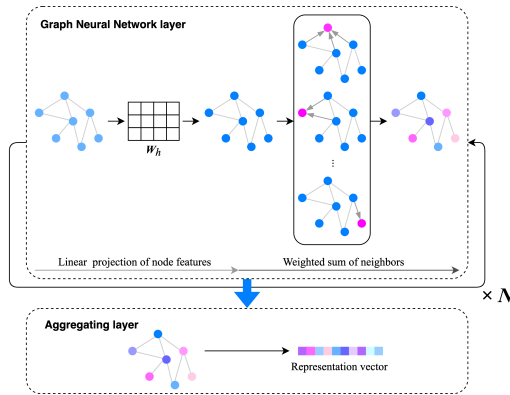


Fig. 2. Conceptualization of a GNN model

In this study, we utilize the power of *Graph Attention Networks* (GAT) (Velickovic *et al.*, 2018), which is the most well-known and widely-used variant of GNNs. Assuming we have a graph  $\mathcal{G} = (V, E)$  and each node has a fixed number of features  $F$ , the input for GAT contains an adjacency matrix  $\mathbf{A}$  and a list of node feature vectors  $\mathbf{X}$  defined as (1) and (2).

$$\mathbf{A}_{ij} = \begin{cases} 1 & \text{if } i \text{ and } j \text{ are connected by an edge} \\ 0 & \text{otherwise} \end{cases} \quad (1)$$

$$\mathbf{X} = \{x_1, x_2, \dots, x_{|V|}\} \text{ with } x_i \in \mathbb{R}^F \quad (2)$$

Firstly, the GAT performs linear projection to put the node feature vectors into  $F'$ -dimensioned embedding space by (3). In (3),  $\mathbf{W}_h \in \mathbb{R}^{F' \times F}$  is a learnable weight matrix, and  $x_i^h$  is the projected  $i$ -th node feature vector.

$$x_i^h = \mathbf{W}_h x_i, \quad i = \overline{1, |V|} \quad (3)$$

Then the GAT calculates the attention coefficients  $e_{ij}$  for all pairs of  $(i, j)$  nodes. These coefficients are then normalized by the *softmax* function to decrease the bias and the cost of computing. When normalizing for  $e_{ij}$ , only nodes which are connected to  $i$ -th node are considered. Finally, the higher representation of each node is produced by weighted sum of its neighbor nodes using attention coefficients. Equation (4a), (4b) and (4c) are formal definitions of above operations.

$$e_{ij} = (x_i^h)^T \mathbf{W}_a x_j^h + (x_j^h)^T \mathbf{W}_a x_i^h, \quad i, j = \overline{1, |V|} \quad (4a)$$

$$a_{ij} = \frac{\exp(e_{ij})}{\sum_{k \in C_i} \exp(e_{ik})} \mathbf{A}_{ij}, \quad i, j = \overline{1, |V|} \quad (4b)$$

$$x'_i = \sum_{j \in C_i} a_{ij} x_j^h, \quad i = \overline{1, |V|} \quad (4c)$$

In (4a),  $e_{ij}$  is the attention coefficient reflecting the importance of  $j$ -th atom to  $i$ -th atom and  $\mathbf{W}_a \in \mathbb{R}^{F' \times F'}$  is a learnable weight matrix. In (4b),  $a_{ij}$  is the normalized attention coefficient corresponding to  $e_{ij}$  and  $C_i$  is the set of neighbor nodes of  $i$ -th node. In (4c),  $x'_i$  is the higher representation of  $i$ -th node feature vector. Then the list  $\mathbf{X}' = \{x'_i \in \mathbb{R}^{F'} | i = \overline{1, |V|}\}$  is the output of the GAT layer.

## 3 The Atom-enhanced Graph Neural Network Model with Multi-hop Gating Mechanism for Drug-Target Interaction Prediction

### 3.1 The baseline model

Our baseline model reuses the study of Lim *et al.* (2019). This is a novel GNN-based model that can integrate the 3D structure into compound and protein representations. Figure 3 illustrates the overview of the model.

#### 3.1.1 Model input representation

Firstly, the model takes the input of a compound and a protein as a graph. According to the original study, while all atoms in the compound are taken into the graph, only protein atoms in the radius of 8Å to any compound atoms are considered. The graph is presented by a matrix of atom features ( $\mathbf{X}$ ) and two adjacency matrix ( $\mathbf{A}^1$ ,  $\mathbf{A}^2$ ) and equations (5), (6) and (7) shows the way to create these matrices, respectively. Figure 3 has visualized a conceptual view of matrix  $\mathbf{X}$ ,  $\mathbf{A}^1$  and  $\mathbf{A}^2$ .

$$\mathbf{X} = \{x_1, x_2, \dots, x_M\} \text{ with } x_i \in \mathbb{R}^F \quad (5)$$

$$\mathbf{A}_{ij}^1 = \begin{cases} 1 & \text{if } i \text{ and } j \text{ are connected by covalent bond or } i = j \\ 0 & \text{otherwise} \end{cases} \quad (6)$$

$$\mathbf{A}_{ij}^2 = \begin{cases} \mathbf{A}_{ij}^1 & \text{if } i, j \in \text{protein or } i, j \in \text{compound} \\ e^{-(d_{ij}-\mu)^2/\sigma} & \text{if } i \in \text{protein and } j \in \text{compound,} \\ & \text{or if } i \in \text{compound and } j \in \text{protein} \\ 0 & \text{otherwise} \end{cases} \quad (7)$$

Table 1. The list of atom features used in original study and in Improvement 1

Feature	Value
<i>Original</i>	
Atom type	C,N,O,S,F,P,Cl,Br,B,H (onehot)
Degree of atom	0, 1, 2, 3, 4, 5, 6 (onehot)
Number of H atoms attached	0, 1, 2, 3, 4 (onehot)
Implicit valence electrons	0, 1, 2, 3, 4, 5 (onehot)
In aromatic	0 or 1
<i>Added in Improvement 1</i>	
Hydrogen D/A	[is_donor, is_acceptor]
Pos/Neg Ionizable	[is_pos, is_neg]
In lumped hydrophobe	0 or 1

In (5),  $x_i$  is a feature vector of an atom which contains  $F$  features shown in Table (1) and  $M$  is the total atoms in the graph representing both compound and protein. In (6) and (7),  $i$  and  $j$  are the atom indexes with the same order as of  $\mathbf{X}$ .  $\mathbf{A}^1_{ij}$  and  $\mathbf{A}^2_{ij}$  are the elements at  $i$ -th row and  $j$ -th column in the  $\mathbf{A}^1$  and  $\mathbf{A}^2$  matrix, corresponding. In (7),  $d_{ij}$  is the distance between  $i$ -th atom and  $j$ -th atom and  $\mu$  and  $\sigma$  are learnable parameters.

To adapt this model for our chosen datasets, we have modified the required input by replacing the 8Å-radius atoms of protein with the protein pocket (protein cavity). The protein pocket is proved to be the place where almost interactions occur (Johnson and Karanicolas, 2013). That means if a compound is considered active with a protein, it creates many strong interactions with protein atoms in the protein cavity. Therefore, using the protein cavity makes much more sense than the original method.

After all the inputs are prepared, they are then passed to model for predicting compound-protein interaction.

### 3.1.2 Model architecture

In this baseline model, Lim and his partners used the GAT layer as the main layer for feature extraction. However, they modified the original GAT layer by adding a gating mechanism at the end of the layer to control how much feature information is passed through. In a formal form, equation (4c) of the original GAT is replaced by (8).

$$\begin{cases} x_i^{temp} = \sum_{j \in C_i} a_{ij} x_j^h, & i = \overline{1, |V|} \\ z_i = \sigma(\mathbf{W}_o(x_i || x_i^{temp}) + b), & i = \overline{1, |V|} \\ x'_i = z_i x_i + (1 - z_i) x_i^{temp}, & i = \overline{1, |V|} \end{cases} \quad (8)$$

In (8),  $\mathbf{W}_o \in \mathbb{R}^{1 \times 2F' \times n}$  is a learnable weight matrix and '||' is the concatenation operator.

Let  $\mathbf{GAT}()$  be the formal representation of all GAT layer formulations which are (3), (4a), (4b) and (8). With the input of  $(\mathbf{X}, \mathbf{A}^1, \mathbf{A}^2)$  created from the complex of protein and compound, we define a GAT block that takes these input and produces the higher representation for  $\mathbf{X}$ . Specifically, the GAT block separates the input into  $(\mathbf{X}, \mathbf{A}^1)$  and  $(\mathbf{X}, \mathbf{A}^2)$ , passes them to GAT layer to get output  $\mathbf{X}'_1$  and  $\mathbf{X}'_2$  and perform subtraction  $\mathbf{X}'_2 - \mathbf{X}'_1$  for model to learn the difference between the structure in a binding pose and the structure as separated. Equation (9) presents insights of a GAT block.

$$\begin{cases} \mathbf{X}'_1 = \mathbf{GAT}(\mathbf{X}, \mathbf{A}^1) \\ \mathbf{X}'_2 = \mathbf{GAT}(\mathbf{X}, \mathbf{A}^2) \\ \mathbf{X}'_{out} = \mathbf{X}'_2 - \mathbf{X}'_1 \end{cases} \quad (9)$$

In (9),  $\mathbf{X}'_{out}$  is the output of GAT block. According to the original study, authors stacked  $N$  GAT block to achieve better feature representations. This can be done by using the output of the previous

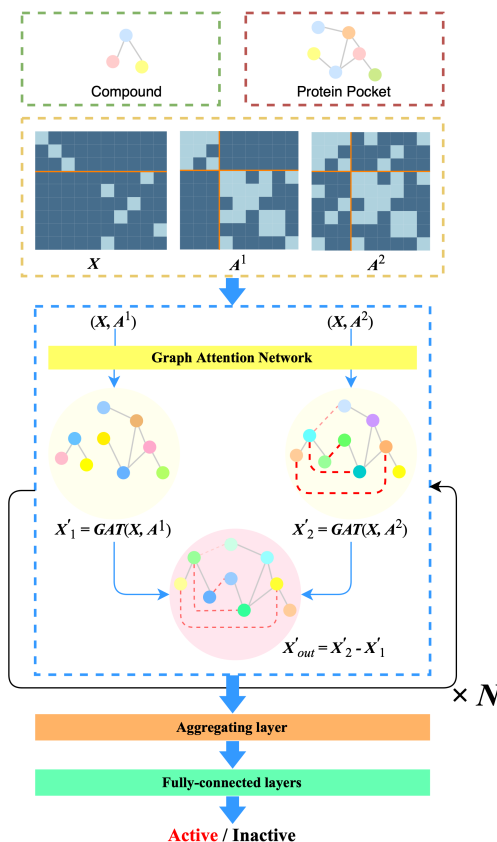


Fig. 3. The architecture of the baseline model

GAT block  $\mathbf{X}'_{out}$  with two adjacency matrixes  $\mathbf{A}^1, \mathbf{A}^2$  as the input for the next GAT block. Please notice that the number of nodes in the GAT layer is equal to the total number of atoms of both protein and compound ( $|V| = M$ ).

The output refined atom feature vectors of the last GAT block are aggregated in the next step to form a feature vector  $x^{complex}$  representing the complex of the input protein and compound. Equation (10) gives the formulation for creating this vector. Finally, a classifier with multiple fully-connected layers is employed to decide if the input complex is active or not. A fully-connected layer is a non-linear transformation that is defined in (11). An overview of the baseline model is also showed in Figure 3.

$$x^{complex} = \sum_{i \in compound} x_i \quad (10)$$

$$y = \sigma(\mathbf{W}_c x + b) \quad (11)$$

In (11),  $x, y$  is the input and output fully-connected layer corresponding. The  $\mathbf{W}_c$  is a learnable weight matrix and  $b$  is the bias. Each fully-connected layer in the classifier has its activation function  $\sigma$  as the *ReLU* function except the *sigmoid* function for the final one.

### 3.2 The transfer learning strategy

As mentioned in the above sections, to deal with insufficient drug data of Coronavirus’s Mpro protein, we apply a transfer learning strategy for learning the general pattern (general interaction rules) before exploring specific rules of Mpro protein. Firstly, for each model with configured settings (e.g improvements), we perform pretraining using the PDBbind dataset for the model to obtain the generalizability. After that, we finetune the pretrained weights of the model using the Fragalysis dataset. With this process, we expect the finetuned model learned both the common interaction rules of any compound-protein and the specific rules for Coronavirus protein (Mpro protein). To demonstrate how transfer learning affects the model performance, we also train the scratch model with the Fragalysis dataset for comparison. Figure 4 shows the overview of the above flows including normal flow (model is directly trained on Fragalysis) and transfer learning flow (model is pretrained before being finetuned on Fragalysis).

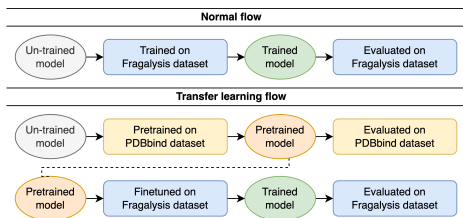


Fig. 4. Illustration of the normal flow and transfer learning flow used in this study

### 3.3 The Atom-enhanced GNN model with Multi-hop Gating Mechanism

In this study, we introduce an improved model from the baseline one, known as *Atom-enhanced GNN model with Multi-hop Gating Mechanism*<sup>1</sup>, in which the following improvements are carried out.

#### 3.3.1 Improvement 1: Enriched atom encoding

The first improvement is enriching atom encoding which adds more chemical features to the representation of each atom. The newly added features include atom degree of six, whether an atom is a hydrogen donor or hydrogen acceptor, whether an atom can be positive or negative ionizable, whether an atom is in any lumped hydrophobe, which are the prerequisites of the corresponding bonding type (a hydrogen bonding requires a hydrogen donor and a hydrogen acceptor atom, so on). Table 1 summarizes all features that are used in this improvement.

#### 3.3.2 Improvement 2: Total atom aggregation

The second improvement basically bases on an assumption that interactions are only created if the protein and the compound match some interaction rules (Lim *et al.*, 2021). Therefore, we have modified the original aggregating layer which calculates the sum of all compound atom vectors to a combination of compound and protein representation vector. With our modification, any protein atoms that have a minimum distance to any compound atoms less than 5Å will be taken into consideration for interaction prediction. The mathematical formulations for our new aggregating layer are proposed in (12a)-(12c).

<sup>1</sup> Our implementation is available at <https://github.com/ViDok-BK/GMGM>

$$x^{complex} = (x^{compound} || x^{protein}) \quad (12a)$$

$$x^{compound} = \sum_{i \in compound} x_i \quad (12b)$$

$$\begin{cases} x^{protein} = \sum_{i \in P} x_i \\ P = \{x_p, p \in protein | \exists c \in compound : dist(p, c) < 5\text{\AA}\} \end{cases} \quad (12c)$$

In (12a), ‘||’ is the concatenation operator and in (12c),  $dist(p, c)$  is the Euclidean distance (O’Neill, 2006) between protein atom  $p$  and compound atom  $c$ .

#### 3.3.3 Improvement 3: Multi-hop gating mechanism

The third improvement is the multi-hop gating mechanism. In this improvement, we repeat the calculation of the gating mechanism multiple times. The reason for this improvement is to enlarge the receptive field of an atom, which is basically based on an assumption in chemistry that atoms having the same function (e.g. hydrophilic, hydrophobic, etc.) usually concentrate together and create a wide area of influence over non-neighbors. Figure 5 gives a more intuitive view of this mechanism of influence. This improvement is inspired by the attention diffusion mechanism, which is proposed by Wang *et al.* (2021). The process of repeating the gating mechanism calculation exactly matches the process of approximate computation for attention diffusion except for the gating coefficient  $z_i$  which is computed from node feature vectors compared to manually input in Wang’s study. Let  $K$  be the number of hops in the receptive field of an atom, the process of calculating multi-hop gating mechanism is presented in Algorithm 1. Please notice that in our proposed improvement, we follow Wang’s study to use  $x_i^h$  for computing  $x_i^{(k)}$ ,  $k = 1, K$  instead of  $x_i$  as in original study of Lim.

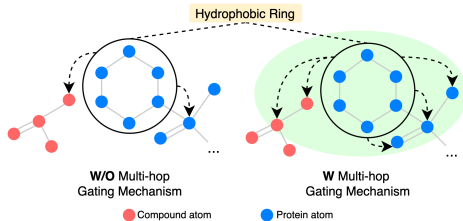


Fig. 5. The difference between models with and without multi-hop gating mechanism

#### Algorithm 1: Multi-hop gating mechanism

**Input** : Normalized attention coefficients  $a_{ij}$ , where  $i, j = \overline{1, |V|}$   
 Atom feature vectors  $x_i^h$ , where  $i = \overline{1, |V|}$   
 Number of hops  $K$   
**Output**: Refined atom feature vectors  $x_i^{(K)}$ , where  $i = \overline{1, |V|}$   
 $x_i^{(0)} = x_i^h, i = \overline{1, |V|}$   
**for**  $k$  in  $Range(1 \dots K)$  **do**  
 $x_i^{temp} = \sum_{j \in C_i} a_{ij} x_j^{(k-1)}, i = \overline{1, |V|}$   
 $z_i = \sigma(\mathbf{W}_o(x_i^{(0)} || x_i^{temp}) + b), i = \overline{1, |V|}$   
 $x_i^{(k)} = z_i x_i^{(0)} + (1 - z_i) x_i^{temp}, i = \overline{1, |V|}$   
**end**  
**return**  $\mathbf{X}^{(K)} = \{x_i^{(K)} | i = \overline{1, |V|}\}$

## 4 Experiments

### 4.1 Datasets and configurations

First of all, we summarize two datasets that are used to train and test our model in Table 2. In more detail, PDBbind dataset is used for pretraining while Fragalysis dataset is used for finetuning our model. We also use the Fragalysis to directly train our model for demonstrating the affect of transfer learning. The PDBbind dataset is splitted into training and testing sets with a ratio of 8:2 while the Fragalysis dataset is splitted with a ratio of 7:3. Table 2 indicates the number of active and inactive compounds in each dataset. A complex of protein and compound is labeled active or inactive based on its IC50. In case IC50 of a complex is equal to or lower than  $2.5\mu M$  it is considered as active and otherwise, it is inactive. Due to the disproportion in the number of active and inactive samples, we implement the undersampling technique to the class with higher quantity to wipe out the bias in training process.

Table 2. The number of compound-protein complexes used for training and testing of each dataset

	PDBbind			Fragalysis		
	Active	Inactive	Total	Active	Inactive	Total
Training	10037	5237	15274	75	125	200
Testing	2530	1287	3817	35	54	89

Table 3. AUC scores of baseline model compared to other models in various settings grouped by molecular representation type

Model with settings	Directly trained on Fragalysis	Pretrained on PDBbind	Finetuned on Fragalysis
<i>String-based representation</i>			
DeepDTA	0.870	<b>0.849</b>	0.862
<i>String-based + Feature matrix representation</i>			
DrugVQA	0.853	0.819	0.820
<i>Graph-based + String-based representation</i>			
GraphDTA-GINConvNet	0.885	0.838	0.874
GraphDTA-GATNet	<b>0.886</b>	0.814	0.890
GraphDTA-GCNNet	0.868	0.836	0.862
GraphDTA-GAT_GCN	0.874	0.835	0.874
<i>Graph-based representation</i>			
Baseline model	0.841	0.758	0.859
Baseline + Ipmt 1	0.865	0.787	0.896
Baseline + Ipmt 2	0.877	0.785	0.915
Baseline + Ipmt 3	0.870	0.793	0.936
Baseline + Ipmt 1,2	0.822	0.813	0.930
Baseline + Ipmt 1,2,3	0.868	0.820	<b>0.938</b>

Before showing the training results, we want to specify the hyperparameters that are used in model implementation. The models are composed of 4 GNN layers, the dimension of each node embedding vector in GNN layer is 140. Our implementation uses the same number of fully connected layers in the final classifier as the original study which is 4 and the dimension of those layers equals 128 except the final one is only one (due to the binary classification problem) (Lim et al., 2019). We use a batch size of 16 in the training process. To avoid overfitting, we also adopt the dropout rate to 0.3. Regarding the training of models that applies Improvement 3, the number of hops is set to 5, which is proven to produce better results according to Wang et al. (2021). The final hyperparameter is the number of epochs, which differ while training on different datasets and

are not dependent on the models. When training directly on Fragalysis, we choose the result with the highest AUC score out of the first 1000 epochs. As for the pretrain model, the best checkpoint among the first 50 epochs will be selected. Then, that checkpoint will be used to finetune on Fragalysis and the best result of the first 1000 epochs will be chosen. All of our experiments are performed on Google Colab with 2-core CPU, Tesla K80 GPU and 12GB of RAM.

To objectively validate our improved model performance, we perform comparisons with other deep learning-based models. Particularly, we choose top-tier models which requires different input representations including DeepDTA (string-based) (Öztürk et al., 2018), DrugVQA (string-based for compound + feature matrix for protein) (Zheng et al., 2020) and GraphDTA (graph-based for compound + string-based for protein) (Nguyen et al., 2020). These methods were reproduced using official implementations from authors and the best results are chosen to report. We use the metric of *Area Under the ROC Curve* (AUC score) (Bradley, 1997) to judge overall performance between models. Table 3 summarizes the AUC score of the above models and our model in different settings and scenarios. The first column indicates the models with their settings. The last three columns store the AUC scores of models when being directly trained on the Fragalysis, pretrained on PDBbind and finetuned on Fragalysis dataset, respectively.

### 4.2 The benefit of transfer learning strategy

In Table 3, we can observe improvements in accuracy when our models have been pre-trained on the PDBbind compared to directly trained on the Fragalysis. Specifically, toward the baseline model, the AUC score increases from 0.841 to 0.859 (increased 0.018). With settings including our improvements, the AUC scores make a big leap, where, with the two first improvements, the difference is significant up to 0.108. This suggests that without the pretraining process, the models generally can't learn how compounds interact with protein and seem overfitted when our improvements are applied. This is clearly seen by the combination of Improvement 1 and 2 only achieves an AUC score of 0.822 which is lower than that of the baseline model (0.841), while in the finetuning scenario, it achieves better performance than the baseline one ( $0.930 > 0.859$ ). Although the results when being pretrained on the PDBbind dataset are not too high, the pretraining process helps our models increase their generalizability and learn the general interaction principles. Therefore, the finetuning results on the Fragalysis are not overfitted and achieve better performance than results of directly trained models.

### 4.3 The effects of proposed improvements

According to Table 3, when applying each improvement separately, the produced results are better than that of the baseline model in all scenarios. In a more insightful view, when applying each improvement separately, the results always have higher AUC scores than that of the baseline model. The most effective improvement belongs to the multi-hop gating mechanism. With this mechanism, finetuned model achieves 0.936 (improved 0.077 from the baseline). Moreover, when our improvements are combined together, they can boost the model performance to a higher level. The combination of Improvement 1 and 2 results in AUC scores of 0.813 and 0.930 which are better than those of single improvement configurations in PDBbind pretraining scenario and in Fragalysis finetuning scenario respectively. Toward the best results our models achieved, the setting including all three improvements reaches an outstanding score of 0.938 in comparison with the scores from other settings in the finetuning scenario. From the above observations, we can consider that the multi-hop gating mechanism is really effective in creating a larger influential field for an atom or group of atoms. Thus, this helps our models better generalize



the functional groups in both protein and compound, which leads to outstanding results.

#### 4.4 Comparison with other methods

When being trained directly on Fragalysis, the combination of the baseline model and Improvement 2 achieves 0.877 AUC score, which is better than that of DeepDTA and DrugVQA, at 0.870 and 0.853, respectively. However, the GraphDTA method has a setting that reaches 0.886, which is higher than all of our models. In case being pretrained on PDBbind, our models show slightly lower performance. Specifically, the highest AUC score of our models is 0.820, which is lower than almost AUC scores of GraphDTA and lower than that of DeepDTA (0.849). In spite of the unsatisfactory results on the PDBbind dataset, our models outperform the others when being finetuned on the Fragalysis dataset. In detail, all settings that have our improvements achieve AUC scores from 0.896 up to 0.938, which are strictly higher than the highest AUC score of GraphDTA (0.890), DeepDTA (0.862), and DrugVQA (0.820). These outstanding results suggest that our improved models are good at learning the general interaction principles and thus, when being finetuned on Fragalysis, they achieve better results than other methods.

## 5 Conclusion

Drug-Target Interaction prediction has been a last-long problem in drug discovery and it makes an essential contribution to the success of developing drugs. Toward the COVID-19 situation, the more effective the method of solving this problem is, the quicker and less cost-consuming the drug development process is. With our proposed model together with the transfer learning strategy, we achieve a noticeable performance compared to the baseline model and other state-of-the-art ones. With these results, our model can be applied in many COVID-19 treatment research centers to boost their productivity in designing candidate drugs. In the future, our improved model can be developed to integrate some weighting functions for assessing the importance of both intra-molecular covalent and inter-molecular non-covalent interactions. Moreover, the loss function can be upgraded for evaluating the strength of pairwise interactions and the GAT layer can be modified to include edge features. In conclusion, this model has much room to improve including both the architecture and the optimization process.

## Acknowledgements

This research is funded by Ho Chi Minh City University of Technology (HCMUT) under grant number SVKSTN-2021-KH&KTMT-38. We acknowledge the support of time and facilities from Ho Chi Minh City University of Technology (HCMUT), VNU-HCM for this study.

## References

Berman, H. M., Westbrook, J., Feng, Z., Gilliland, G., Bhat, T. N., Weissig, H., Shindyalov, I. N., and Bourne, P. E. (2000). The Protein Data Bank. *Nucleic Acids Research*, **28**(1), 235–242.

Bradley, A. P. (1997). The use of the area under the ROC curve in the evaluation of machine learning algorithms. *Pattern Recognition*, **30**(7), 1145–1159.

Burki, T. K. (2022). The role of antiviral treatment in the COVID-19 pandemic. *The Lancet Respiratory Medicine*, **10**(2), e18.

Chodera, J., Lee, A. A., London, N., and von Delft, F. (2020). Crowdsourcing drug discovery for pandemics. *Nature Chemistry*, **12**(7), 581–581.

Diamond (2021). Fragalysis. <https://fragalysis.diamond.ac.uk>. Accessed on Oct. 2021.

Gao, K. Y., Fokoue, A., Luo, H., Iyengar, A., Dey, S., and Zhang, P. (2018). Interpretable drug target prediction using deep neural representation. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18*, pages 3371–3377. International Joint Conferences on Artificial Intelligence Organization.

Johnson, D. K. and Karanikolas, J. (2013). Druggable Protein Interaction Sites Are More Predisposed to Surface Pocket Formation than the Rest of the Protein Surface. *PLoS Computational Biology*, **9**(3), 1–10.

Keshavarzi Arshadi, A., Webb, J., Salem, M., Cruz, E., Calad-Thomson, S., Ghadirian, N., Collins, J., Diez-Cecilia, E., Kelly, B., Goodarzi, H., and Yuan, J. S. (2020). Artificial Intelligence for COVID-19 Drug Discovery and Vaccine Development. *Frontiers in Artificial Intelligence*, **3**.

Koes, D. R., Baumgartner, M. P., and Camacho, C. J. (2013). Lessons Learned in Empirical Scoring with smina from the CSAR 2011 Benchmarking Exercise. *Journal of Chemical Information and Modeling*, **53**(8), 1893–1904.

Kontopantelis, E., Mamas, M. A., Webb, R. T., Castro, A., Rutter, M. K., Gale, C. P., Ashcroft, D. M., Pierce, M., Abel, K. M., Price, G., Faivre-Finn, C., Van Spall, H. G. C., Graham, M. M., Morciano, M., Martin, G. P., Sutton, M., and Doran, T. (2022). Excess years of life lost to COVID-19 and other causes of death by sex, neighbourhood deprivation, and region in England and Wales during 2020: A registry-based study. *PLoS Medicine*, **19**(2), 1–20.

Lee, I., Keum, J., and Nam, H. (2019). DeepConv-DTI: Prediction of drug-target interactions via deep learning with convolution on protein sequences. *PLoS Computational Biology*, **15**(6), 1–21.

Lim, J., Ryu, S., Park, K., Choe, Y. J., Ham, J., and Kim, W. Y. (2019). Predicting Drug-Target Interaction Using a Novel Graph Neural Network with 3D Structure-Embedded Graph Representation. *Journal of Chemical Information and Modeling*, **59**(9), 3981–3988.

Lim, S., Lu, Y., Cho, C. Y., Sung, I., Kim, J., Kim, Y., Park, S., and Kim, S. (2021). A review on compound-protein interaction prediction methods: Data, format, representation and model. *Computational and Structural Biotechnology Journal*, **19**, 1541–1556.

Muegge, I. and Mukherjee, P. (2016). An overview of molecular fingerprint similarity search in virtual screening. *Expert Opinion on Drug Discovery*, **11**(2), 137–148.

Nguyen, T., Le, H., Quinn, T. P., Nguyen, T., Le, T. D., and Venkatesh, S. (2020). GraphDTA: predicting drug-target binding affinity with graph neural networks. *Bioinformatics*, **37**(8), 1140–1147.

O'Neill, B. (2006). Chapter 2 - frame fields. In B. O'Neill, editor, *Elementary Differential Geometry (Second Edition)*, pages 43–99. Academic Press, Boston, second edition edition.

Riley, K. E. and Hobza, P. (2011). Noncovalent interactions in biochemistry. *WIREs Computational Molecular Science*, **1**(1), 3–17.

Scarselli, F., Gori, M., Tsoi, A. C., Hagenbuchner, M., and Monfardini, G. (2009). The Graph Neural Network Model. *IEEE Transactions on Neural Networks*, **20**(1), 61–80.

Thafar, M., Raies, A. B., Albaradei, S., Essack, M., and Bajic, V. B. (2019). Comparison Study of Computational Prediction Tools for Drug-Target Binding Affinities. *Frontiers in Chemistry*, **7**.

Thafar, M. A., Olayan, R. S., Ashoor, H., Albaradei, S., Bajic, V. B., Gao, X., Gojobori, T., and Essack, M. (2020). DTiGEMS+: drug-target interaction prediction using graph embedding, graph mining, and similarity-based techniques. *Journal of Cheminformatics*, **12**(1), 44.

Trott, O. and Olson, A. J. (2010). AutoDock Vina: Improving the speed and accuracy of docking with a new scoring function, efficient optimization, and multithreading. *Journal of Computational Chemistry*, **31**(2), 455–461.

Velickovic, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., and Bengio, Y. (2018). Graph attention networks. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net.

Wang, G., Ying, R., Huang, J., and Leskovec, J. (2021). Multi-hop attention graph neural networks. In Z.-H. Zhou, editor, *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*, pages 3089–3096. International Joint Conferences on Artificial Intelligence Organization. Main Track.

Wang, J. and Dokholyan, N. V. (2019). MedusaDock 2.0: Efficient and Accurate Protein-Ligand Docking With Constraints. *Journal of Chemical Information and Modeling*, **59**(6), 2509–2515.

WHO (2021). Coronavirus. <https://www.who.int/health-topics/coronavirus>. Accessed on Jan. 2022.

Zheng, S., Li, Y., Chen, S., Xu, J., and Yang, Y. (2020). Predicting drug-protein interaction using quasi-visual question answering system. *Nature Machine Intelligence*, **2**(2), 134–140.

Öztürk, H., Özgür, A., and Ozkirimli, E. (2018). DeepDTA: deep drug-target binding affinity prediction. *Bioinformatics*, **34**(17), 1821–1829.