Deep Learning-Enhanced Drug Discovery: Innovative Molecule Clustering and Interaction Prediction through Graph Analysis

by

Hamid Hadipour

A Thesis submitted to the Faculty of Graduate Studies of

The University of Manitoba

in partial fulfillment of the requirements of the degree of

MASTER OF SCIENCE

Department of Computer Science

University of Manitoba

Winnipeg

Thesis advisors
Dr. Pingzhao Hu & Dr. Carson K. Leung

Author
Hamid Hadipour

Deep Learning-Enhanced Drug Discovery: Innovative Molecule Clustering and Interaction Prediction through Graph Analysis

Abstract

Motivation

The quest for efficient drug discovery processes necessitates a comprehensive approach that integrates molecular feature analysis with accurate compound-protein interaction (CPI) prediction. This study introduces models that combine deep learning (DL) techniques for intricate molecular feature engineering and innovative CPI prediction methods. This integration responds to the need for detailed molecular dataset analysis and the prediction of interactions between novel compounds and proteins, thereby enhancing drug discovery.

Methods and Results

Chapter 3 - Molecular Clustering and Feature Analysis:

The framework implements a feature engineering scheme focusing on molecule-specific atomic and bonding information. It utilizes principal component analysis (PCA) for encoding this information and a variational autoencoder (VAE)-based method for embedding both global chemical properties and local features. This approach facilitated the clustering of a large dataset containing over 47,000 molecules. Using the K-means method with 32 embedding`s size based on the VAE method, 50 distinct molecular clusters were identified. These clusters were visualized

through t-distributed Stochastic Neighbor Embedding (t-SNE), showcasing the framework's capability in effectively grouping molecules based on their complex features.

Chapter 4 - CPI Prediction with GraphBAN:

For CPI prediction, the study introduces GraphBAN, a novel inductive-based approach using graph knowledge distillation (KD). This component incorporates a deep bilinear attention network (BAN) and a KD module for graph analysis, enabling the alignment of interaction features across different distributions. GraphBAN's functionality extends to both transductive and inductive link predictions in a bi-partite graph of CPIs. Tested against three benchmark datasets, GraphBAN demonstrated superior performance, outperforming six baseline models. It shows that it is able to predict interactions between unseen compounds and proteins that is an important aspect of drug discovery.

Conclusion

This study presents two innovative models that parallelly analyze molecule-specific feature engineering and advanced CPI prediction techniques. By integrating these two key components, the models not only deepen the understanding of molecular characteristics but also significantly boost the accuracy of CPI predictions. This advancement is crucial for streamlining drug discovery processes, reducing the number of compounds needed for screening, and facilitating the development of more effective and targeted drugs.

My Contribution to Thesis

Chapter 3 of my thesis, a significant part of my research, was collaboratively developed with my lab mate, Chengyou Liu. This chapter has been published as an article: [Hadipour H, Liu C, Davis R, Cardona ST, Hu P. Deep clustering of small molecules at large-scale via variational autoencoder embedding and K-means. BMC bioinformatics. 2022 Apr;23(4):1-22]. I would like to express my gratitude to Chengyou Liu for his substantial contributions, particularly his expertise in cheminformatics techniques, which were crucial in formulating the core ideas and arguments of this chapter.

My Specific Contributions: In this collaborative effort, my primary focus was on the development and application of various clustering methods and performance metrics. My contributions are detailed in the following sections of Chapter 3:

- 3.2.6 Molecule Clustering
- 3.2.7 K-Means Method
- 3.2.8 BIRCH Method
- 3.2.9 DL Autoencoder-based K-means Clustering
- 3.2.10 DL Variational Autoencoder-based K-means Clustering
- 3.2.11 Estimation of the Number of Molecule Clusters
- 3.2.12 Clustering Performance Evaluation
- 3.2.13 Calinski-Harabasz Index
- 3.2.14 Davies-Bouldin Index
- 3.2.16.1 t-SNE Visualization of the Molecular Embeddings

- 3.3.1 Estimating the Number of Clusters
- 3.3.2 Performance Evaluation of the Identified Molecule Clusters
- 3.3.3 Visualization of the Identified Clusters (except Figures 3-9, 3-10)

Through these contributions, I have endeavored to advance the methodology for analyzing the diversity of small molecule datasets on a large scale.

Acknowledgements

I would like to express my sincere gratitude to my supervisor Dr. Pingzhao Hu for his consistent support and excellent guidance throughout my M.Sc. journey at the University of Manitoba. Over the past two years, Dr. Hu has been providing me with countless hours of support and guidance when I encountered any obstacles or confusion, both academically and personally, for which I am deeply grateful.

I would like to extend my gratitude to Dr. Silvia Cardona and Dr. Rebecca Davis for their support, feedback, and encouragement. It was an honor for me to have this valuable collaboration experience in such an outstanding interdisciplinary team. Furthermore, I would also like to express my thank to Dr. Carson K. Leung for his co-supervision and my advisory/examining committee members Dr. Cuneyt Akcora (Computer Science) and Dr. Silvia Cardona (Microbiology) for their insightful comments on my thesis. Thanks Dr. Celine Latulipe for chairing the M.Sc. defence.

Special thanks to all my colleagues and friends at the Hu Lab for their support and help: Qian Liu, Mohd Wasif Khan, Md. Mohaiminul Islam, Yan Sun, Chengyou Liu, Saqib Islam, Leann Lac, Daryl Fung, and Judah Zammit. I am also grateful to the members from the labs of Dr. Silvia Cardona and Dr. Rebecca Davis: Andrew M. Hogan, A. S. M. Zisanur Rahman, Hunter Sturm, and Julieta Novomisky Nechcoff. Additionally, I would like to thank all the faculty and staff members in the Department of Computer Science who have supported me in various ways.

Last but not least, I would like to dedicate this thesis to my partner, Golnaz. Her priceless love and emotional support helped me get through the challenging times and inspired me to strive for

excellence. Her unwavering faith in my abilities and constant encouragement were the cornerstones of my journey, making this achievement as much hers as it is mine.

Contents

List of Figures

List of Tables

List of Abbreviations

Abbreviations	Description
CPI:	Compound-protein interaction
BAN:	Bilinear attention network
ML:	Machine learning
DL:	Deep learning
DTI:	Drug target interaction
GNN:	Graph neural network
GAE2A:	Graph autoencoder to attention
KD:	Knowledge distillation
GAE2F:	Graph autoencoder to feature
GAE:	Graph autoencoder
DNN:	Deep neural network
ReLU:	Rectified linear unit
SMILES:	Simplified molecular input line entry system
BCE loss:	Binary cross entropy loss
MLP:	Multilayer perceptron
MSE:	Mean squared error
GCN:	Graph convolutional network
FCFP:	Functional class fingerprints
CNN:	Convolutional neural network
CDAN:	Conditional domain adaptation network
AUROC:	Area under the receiver operating characteristic curve
RF:	Random forest
PSC:	Pseudo amino acid compositions
AUPRC:	Area under the precision recall curve
PCA:	Principal Component Analysis
AE:	Auto Encoder
VAE:	Variational Auto Encoder

t-SNE:	t-distributed stochastic neighbor embedding
MOA:	Mechanism of Action
BIRCH:	Balanced Iterative Reducing and Clustering using Hierarchies
PC:	Principal Component
QSPR:	Quantitative Property Structure Relationship
QSAR:	Quantitative Activity Structure Relationships
HTS:	High Throughput Screening
ECFP:	Extended Connectivity Fingerprint
LSTM	Long-term short-term memory

Chapter 1 Background

1.1 Machine Learning in Drug Discovery

Machine Learning (ML) methods have revolutionized the field of drug discovery by leveraging pattern recognition algorithms to analyze empirical observations of small molecules. It is crucial as this type of analysis reduce the all the costs such as financial costs to human resources needs drastically. These methods discern mathematical relationships and extrapolate them to predict the chemical, biological, and physical properties of new compounds. Various ML algorithms, such as random forest, naive Bayesian, support vector machine, and deep learning (DL), are employed in this process. These approaches are instrumental in predicting the structure of drug targets, identifying, and optimizing potential drug candidates ("hits"), exploring the biological activity of new ligands, and designing models to predict the pharmacokinetic and toxicological properties of drug candidates (1).

The development of Deep Learning (DL) is crucial for transforming medical information into practical, reusable methods. This transformation is not based on hypothetical improvements but on concrete applications in drug design, including the enhancement of ML techniques and the collection of pharmacological data (2). One of the significant advancements in this domain is the development of techniques like DeepBAR, which combine chemistry and DL to calculate the binding affinities between drug candidates and their targets rapidly. Furthermore, the application of DL in drug discovery and development is pivotal due to the complexity and length of the drug development pipelines, which depend on numerous factors. DL provides a set of tools that can

enhance both the discovery process and decision-making in drug development. Notable success stories in the application of DL and ML in drug development have demonstrated the effectiveness and potential of these technologies in advancing medical research and healthcare (3,4).

In summary, the use of DL and ML in drug discovery represents a significant leap in the efficiency and effectiveness of developing new pharmaceuticals. It streamlines the process of identifying and optimizing drug candidates, predicting their properties, and accelerating the overall drug development process. This advancement is critical in an industry where time and accuracy are paramount, potentially leading to faster, more efficient development of life-saving drugs.

1.2 Diversity of Molecule Datasets

Analyzing the diversity of a dataset of molecules at a large scale is a critical step in the field of ML, particularly in applications related to drug discovery, materials science, and chemical engineering (5). This process involves examining a broad array of chemical compounds to understand their structural, functional, and property variations. That is important in terms of enhancing the generalizability of ML models. A diverse dataset ensures that the model is exposed to a wide range of molecular features, enabling it to make accurate predictions for unseen compounds (6,7). By comprehensively characterizing the diversity in the dataset, researchers can reduce biases and improve the robustness of their models, ultimately accelerating the development of novel drugs, materials, and chemical solutions. Furthermore, it aids in the discovery of hidden patterns and relationships between molecular structures, thus advancing our understanding of chemistry and facilitating innovation in various scientific and industrial domains.

1.3 Compound-Protein Interction Analysis

Analyzing CPIs is a crucial aspect of drug discovery, as it involves understanding how chemical compounds, often referred to as small molecules, interact with specific proteins in the cell body (8). Small molecules are low molecular weight organic compounds that can regulate biological processes, often used in pharmacology for drug development. Analyzing these interactions is crucial for the development of new drugs.

Identification of Potential Drug Targets: In drug discovery, one of the aims is to identify proteins (e.g., enzymes, receptors, or signaling molecules) that are associated with diseases (9). Analyzing CPIs helps identify potential drug targets, as compounds that interact with disease-related proteins may offer therapeutic benefits.

Screening for Drug Candidates: Scientists use various techniques, such as high-throughput screening and computational modeling, to test many compounds against a specific protein target (9). By analyzing the interactions between compounds and proteins, potential drug candidates can be identified based on their ability to bind to the target protein and modulate its activity.

Optimizing Drug Design: Understanding the nature of CPIs is crucial for optimizing drug design (10). It allows researchers to fine-tune the chemical structure of a compound to enhance its binding affinity and specificity to the target protein. This optimization leads to the development of more potent and selective drugs with fewer side effects.

Predicting Drug Efficacy and Safety: Analyzing CPIs can help predict the efficacy and safety of a drug candidate (11). By studying how a compound interacts with its target protein and other off-target proteins, researchers can anticipate potential side effects and make informed decisions about drug development.

Repurposing Existing Compounds: Sometimes, existing compounds that were originally developed for one purpose can be repurposed for another (12). Analyzing CPIs can reveal new therapeutic uses for known compounds by identifying interactions with different protein targets.

Reducing Cost and Time in Drug Development: By gaining insights into CPIs early in the drug discovery process, researchers can eliminate fewer promising candidates and focus their efforts and resources on the most promising ones. This reduces the time and cost associated with drug development.

In summary, analyzing CPI is a fundamental step in drug discovery that helps in identifying potential drug targets, optimizing drug design, predicting drug efficacy and safety, and accelerating the development of novel drugs for various diseases.

1.4 Molecule Representation

1.4.1 Simplified Molecular Input Line Entry System

Simplified Molecular Input Line Entry System (SMILES) is a standardized notation system employed in the field of chemistry to succinctly represent chemical structures using alphanumeric characters and symbols that introduced by Weininger in 1988 (13). It functions as a textual representation of molecules, simplifying the complex process of conveying molecular structures without the need for detailed graphical depictions. In SMILES notation, characters are represented by their respective chemical symbols (e.g., 'C' for carbon, 'O' for oxygen), and the connections between atoms are indicated through the use of lines and numerical indices. This notation system serves as an indispensable tool for chemists and researchers, facilitating the unambiguous communication of complex molecular information and fostering collaboration in chemical research, drug

development, and related domains. SMILES provides a common language for representing chemical structures, allowing the automation of various computational tasks. Algorithms can parse and process SMILES strings to analyze chemical properties, calculate molecular descriptors, and predict chemical reactions. This simplifies the automation of tasks such as virtual screening of compound libraries, determining the properties of new molecules, or simulating chemical reactions. Computational chemists can efficiently manipulate and compare vast sets of chemical data, leading to more informed decision-making in drug development.

1.4.2 Molecule Features

In the domain of computational chemistry and ML tasks involving molecules, the molecules represent with SMILES notation and there are different approaches to extract molecule features from the SMILES representation. Here five main computational methods listed.

1.4.2.1 Fingerprint Methods

These are binary or count vectors representing the presence or absence of certain substructures in the molecule. Examples include MACCS keys, extended-connectivity fingerprints (ECFP), and functional class fingerprints (FCFP). These methods are widely used for similarity searching, virtual screening, and compound classification (14).

1.4.2.2 Descriptor Calculation

These are numerical values that describe the molecular properties and are derived from the molecular structure. They include physicochemical properties, topological descriptors, and

quantum-chemical properties. Software like RDKit (15) and Dragon (16) are commonly used for descriptor calculation (17).

1.4.2.3 Graph-Based Methods

In this approach, molecules are represented as graphs where atoms are nodes and bonds are edges. Techniques like graph convolutional networks (GCNs) can be used to extract features from these graphs for ML models (18).

1.4.2.4 Sequence-Based Methods

Some methods treat SMILES strings as sequences and apply natural language processing (NLP) techniques, such as recurrent neural networks (RNNs) and transformers, to capture the sequential information in the SMILES (19).

1.4.2.5 3D Structure Generation and Analysis

Some methods involve generating the 3D structure from SMILES and then extracting features based on the three-dimensional conformation. This can include geometric, steric, and electrostatic properties (20).

Each of these methods has its strengths and applications, and the choice of method depends on the requirements of the study.

1.5 Protein Representation

1.5.1 Amino Acid Sequences

Amino acid sequences represent the fundamental constituents of proteins in a human-readable format. Proteins consist of chains of amino acids, with the specific sequence of these amino acids determining the unique three-dimensional structure of the protein (21). This structural conformation, in turn, dictates the protein's biological function within an organism. Amino acid sequences are determined by the linear arrangement of distinct amino acids, each represented by a specific letter code (e.g., 'A' for alanine, 'L' for leucine). The order and composition of amino acids within a protein sequence are pivotal in specifying its biological activity, as alterations in sequence can result in diverse functions, interactions, and structural configurations.

In bringing biology to ML tasks, amino acid sequences hold significant importance, particularly within the fields of bioinformatics and computational biology. Their principal utility lies in predicting protein properties and functions. ML algorithms can be trained to discern intricate patterns within amino acid sequences and correlate them with various protein attributes, including enzymatic activity, subcellular localization, or associations with diseases (22). These predictive models play a vital role in drug discovery and protein engineering. By observing extensive datasets containing amino acid sequences and their corresponding attributes, ML models can make precise predictions, thereby offering profound insights into the expansive domain of proteins (23). Such advancements contribute to critical developments in fields like personalized medicine and pharmaceutical research.

1.5.2 Protein Features

Protein sequences are vital biological entities, and extracting numerical features from these sequences is crucial for various bioinformatics applications, including protein structure prediction, and CPI analysis. Although there are many different approaches to generate protein features here, we listed the most useful methods suitable for ML tasks.

1.5.2.1 Sequence-Based Features

k-mer Composition: This approach counts the frequency of k-length subsequences in the protein sequence. It was highlighted in a study where various feature extraction methods, including k-mer natural vectors, were compared for constructing phylogenetic trees (24).

Pseudo Amino Acid Composition (PseAAC): Because the simple linear sequence of amino acids doesn't always capture the full complexity of protein characteristics to follow up this limitation, we can use PseAAC. Th PseAAC incorporates additional information about the protein's properties such as hydrophobicity, hydrophilicity, pKa values of amino acids. This method has been widely used and is supported by tools like Pse-in-One 2.0, which covers different modes to obtain protein feature vectors based on pseudo components (25).

1.5.2.2 Physicochemical Properties

Amino Acid Composition (AAC): AAC calculates the proportion of each amino acid type in the protein sequence, as discussed in a study on feature extraction for phosphorylation site detection (26).

Composition, Transition, and Distribution (CTD): This method represents amino acid physicochemical features of sequences and has been used for DNA-binding protein prediction (25).

1.5.2.3 Structural Features

Position-Specific Scoring Matrix (PSSM): This method is a tool for representing the features of a protein in a way that captures both the conservation and the variability of amino acids at specific positions in the protein sequence. Tools like POSSUM generate numerical sequence feature descriptors based on PSSM profiles (27).

1.5.2.4 Deep Learning-based Features (NLP-based)

Word2Vec: This method, originating from natural language processing, involves treating segments of amino acid sequences as "words" to generate vector representations that capture the context within sequences (28).

Transformers: Transformer models, which have revolutionized natural language processing, can be adapted for protein sequences. They are particularly effective in capturing long-range dependencies and complex patterns in sequences (29).

Convolutional neural networks (CNNs): CNNs used to automatically and adaptively learn spatial hierarchies of features from amino acid sequences. CNNs can be particularly effective in extracting local and positionally invariant features, making them useful for tasks like secondary structure prediction or motif recognition (30).

1.6 Graph Neural Networks

Graph Neural Networks (GNNs) are a type of neural network designed for processing graph-structured data. Unlike traditional neural networks that handle fixed-size inputs like images or text, GNNs are adept at handling data represented in graphs. This includes social networks, molecule structures, or any other form of data where the relationships between entities are as important as the entities themselves. GNNs achieve this by focusing on the nodes and edges in a graph, allowing them to capture the complex patterns and dependencies in the data (31).

In drug discovery, the complexity of molecular structures, biochemical interactions, and biological processes can be effectively captured and analyzed using GNNs. These networks have the capability to extract valuable information from chemical and biological graphs, leading to groundbreaking advances in the identification of novel drug candidates, understanding protein-ligand interactions, and optimizing drug development processes. Here, we will explore some of the key points that GNNs can handle in drug discovery, specifically in CPI prediction.

1.6.1 Graph Link Prediction

Graph link prediction is a fundamental task in network analysis, especially in the context of graph data structures. It involves predicting the likelihood of the existence of edges (links) between nodes in a graph. This task is essential for various applications, biological network analysis that we use in this research (32).

The key concepts in graph link prediction include:

1. **Node Embeddings:** Node embeddings are low-dimensional vector representations of nodes in a graph. They capture the structural and topological information of nodes and their neighborhood.

2. **Feature Engineering:** In addition to node embeddings, link prediction models may utilize various node and edge features, such as attributes of nodes and the type of relationships between them. Feature engineering is crucial to improve prediction accuracy.

3. **Scoring Function:** Link prediction models typically employ a scoring function to assess the likelihood of a link between two nodes. Common scoring functions include the dot product, cosine similarity, or more complex functions like the logistic sigmoid or neural network-based approaches.

4. **Negative Sampling:** To train a link prediction model, negative samples (pairs of nodes without a direct edge) are often used in conjunction with positive samples (pairs of nodes with an existing edge) to learn the model parameters.

5. **Evaluation Metrics:** Common evaluation metrics for link prediction include precision, recall, F1-score, and the area under the ROC curve (AUROC), among others. These metrics help assess the model's ability to discriminate between existing and non-existing links.

6. **Challenges:** Challenges in graph link prediction include handling imbalanced data, addressing the cold start problem (predicting links for new unseen nodes), and dealing with large-scale graphs efficiently. As an unexplored path in drug discovery field, in this research study we focus on following the cold start link prediction with GNNs. To explain more about this challenge, we introduce the related key concepts by defining the meanings of transductive, semi-inductive and fully inductive link prediction.

1.6.2 Transductive Link Prediction

Transductive link prediction approaches assume that the graph is fixed, and the prediction is made for a specific set of edges or interactions that are already present in the graph. In the case of compound-protein link prediction, this means that the method predicts interactions only for compounds and proteins that are already known in the graph. These methods leverage information from the existing graph structure and attributes of nodes to make predictions, but they are limited to the nodes present in the original graph.

1.6.3 Semi-inductive Link Prediction

Semi-inductive methods aim to strike a balance between transductive and inductive approaches. They assume a partially known graph and partially unknown nodes. In the context of compound-protein link prediction, this could mean predicting interactions for compounds and proteins that are not present in the original graph, but still leveraging information from known nodes and edges. Semi-inductive methods often rely on techniques like node feature propagation to make predictions for both known and unknown nodes.

1.6.4 Fully Inductive Link Prediction

Fully inductive link prediction methods are designed to make predictions for entirely new nodes and edges do not present in the original graph. In the case of compound-protein link prediction, this would involve predicting interactions for compounds and proteins that were not part of the initial graph. Fully inductive methods are more challenging because they need to learn from the graph's structure and node attributes without direct access to the target nodes.

1.6.5 GraphSAGE

GraphSAGE (Graph Sample and Aggregating) is an innovative framework designed for generating low-dimensional vector representations for nodes in a graph. This method, pivotal in the realm of graph-based learning, particularly shines in its ability to handle large-scale graphs efficiently. Unlike traditional graph neural networks that necessitate the entire graph structure for training, GraphSAGE adopts a novel approach that enables learning from a sampled subset of nodes, significantly enhancing scalability.

At its core, GraphSAGE employs a neighborhood sampling technique combined with an aggregation function to generate embeddings for nodes. The process can be mathematically detailed as follows:

Node Sampling: For each target node v in the graph, a fixed-size sample of its neighbors $N(v)$ is selected. This sampling is crucial as it addresses the challenge of varying neighborhood sizes across nodes in the graph. The sampling size is denoted as k, where k is a hyperparameter that can be tuned based on the graph's properties.

Aggregation Function: GraphSAGE introduces several aggregation functions such as mean, long-term short-term memory (LSTM), and pooling, to aggregate features from the sampled neighborhood. The choice of function can be tailored to the specific requirements of the graph and the learning task. Mathematically, for a node v, the aggregation function AGG operates over its neighbors' features $\{h_u, \forall u \in N(v)\}$ to produce a vector representation $h_{N(v)}$ The aggregated feature $h_{N(v)}$ captures the collective information of the neighborhood.

Feature Update: The node representation is then updated by combining its own features with the aggregated neighborhood features. Formally, the updated feature h'_v for a node v is computed as:

$$h'_v = \sigma(W.CONCAT(h_v, h_{N(v)})), \text{ (1-1)}$$

Here, σ denotes a non-linear activation function, W is a learnable weight matrix, and $CONCAT$ represents the concatenation operation.

Recursive Application: This process is recursively applied for a fixed number of iterations or layers, allowing the model to capture information from larger neighborhoods progressively.

Objective Function: The training of GraphSAGE is guided by an objective function tailored to the specific task, such as node classification or link prediction. This objective function is optimized using stochastic gradient descent or similar optimization techniques.

GraphSAGE's ability to learn from a subset of the graph and its flexible aggregation mechanism makes it a powerful tool for various applications in graph analytics and ML on graphs. By efficiently handling large-scale graphs and adapting to different graph structures, it offers a significant advancement in the field of graph representation learning.

1.7 Transformer Neural Network

The Transformer neural network, introduced in the paper "Attention is All You Need" by Vaswani et al. in 2017 (33), revolutionized the field of DL with its parallelized and scalable approach to sequence modeling which crucial in analyzing complex structures such as protein sequences and chemical structures. It is the foundation of several modern language models, including BERT, GPT, and more.

1.7.1 Self-Attention Mechanism

The self-attention mechanism is the heart of the Transformer. It allows the model to weigh the importance of different elements in an input sequence, considering the relationships and dependencies between them. This mechanism enables the model to understand the context, which is crucial for various tasks.

1.7.2 Multi-Head Attention

Multi-head attention involves multiple sets of self-attention mechanisms working in parallel. It allows the model to focus on different aspects of the input sequence simultaneously, capturing various types of information and improving its ability to learn complex patterns.

1.7.3 Positional Encoding

Positional encoding as an embedded encoder is added to the input`s embedding matrix to provide the Transformer model with information about the order of elements in a sequence. This is necessary for understanding the sequence's structure and complex hidden information.

1.7.4 Encoder-Decoder Architecture

The Transformer comprises two main parts: the Encoder and the Decoder. The Encoder processes the input data, while the Decoder generates the output. This architecture is particularly effective for tasks such as machine translation, where it converts text from one language to another.

1.8 Transformer Networks in Drug Discovery

Transformer networks have shown remarkable success in various fields, including drug discovery, and their application in the domain of molecular biology and bioinformatics is an area of active research (34–36).

Further added how they can be applied in molecule feature extraction, protein feature extraction, and compound-protein feature mapping.

1.8.1 Molecule Feature Extraction

Transformer networks can be effectively used for molecule feature extraction. They can learn complex molecular representations by capturing the relationships between atoms in a molecule. This is achieved by treating each atom and bond as elements in a sequence, similar to words in a sentence in NLP. The self-attention mechanism in the Transformers allows the model to weigh the importance of different atoms and bonds in a molecule, leading to a detailed understanding of molecular structure and properties. This approach is useful for predicting molecular properties and activities. A notable reference in this area is "Molecular Representation Learning with Transformers" by Maziarka et al., 2020, (37) which demonstrates the use of Transformers for molecular feature extraction.

1.8.2 Protein Feature Extraction

In the context of protein feature extraction, the Transformers can analyze amino acid sequences to predict protein structures and functions. Each amino acid in a protein sequence can be considered analogous to a word in a sentence, allowing the Transformer to learn the contextual relationships between amino acids. This capability is essential for understanding protein folding, interactions,

and functional annotations. An remarkable work in this area is the application of the AlphaFold system by Jumper et al., 2021, (38) which utilizes a DL approach, including transformer models, to predict protein structures with high accuracy.

1.8.3 Compound-Protein Feature Mapping

Transformer networks can also be applied to compound-protein feature mapping, a critical task in CPI prediction. By understanding the interaction between compounds (small molecules) and proteins, the Transformers can help predict how a drug will bind to its target protein. This involves learning the complex relationships between the molecular features of the compound and the amino acid sequences of the protein. The self-attention mechanism in the Transformers is particularly useful here, as it can identify the most relevant parts of both the compound and the protein for predicting their interaction. A reference in this area is "Predicting Drug–Target Interaction Using a Novel GNN with 3D Structure-Embedded Graph Representation" by Nguyen et al., 2020 (39).

Chapter 2 Motivation and Research

2.1 Introduction

The rapidly evolving field of drug discovery and development has been significantly influenced by advancements in ML and computational methods. These innovations have opened new avenues for exploring CPI, a critical aspect of pharmaceutical research. This chapter delves into the motivation behind using diverse datasets of small molecules for ML purposes and the development of a model for CPI to facilitate inductive link prediction in drug discovery.

2.2 Motivation

2.2.1 Diversity in Small Molecule Datasets

The diversity of small molecules in a dataset is crucial for developing robust and accurate ML models in drug discovery. A diverse dataset ensures a comprehensive representation of the chemical space, enabling the models to capture a wide range of molecular features and interactions. This diversity is fundamental for several reasons:

Generalizability: Models trained on diverse datasets are more likely to generalize well to unseen compounds, reducing the risk of overfitting to specific chemical structures.

Predictive Accuracy: Diverse datasets improve the predictive accuracy of models by providing a broad spectrum of molecular interactions and properties.

Innovation in Drug Discovery: Exploring a diverse chemical space can lead to the discovery of novel compounds with unique therapeutic potentials.

2.2.2 CPI for Inductive Link Prediction

The second objective focuses on developing a model for CPI, an essential step in understanding the biological activities of compounds. CPI models are vital for:

Target Identification: Identifying the interaction between compounds and proteins helps in understanding the mechanism of action, leading to more effective drug design.

Inductive Link Prediction: By predicting new interactions, CPI models contribute significantly to inductive link prediction, facilitating the discovery of potential drug candidates.

Accelerating Drug Discovery: Efficient CPI models can significantly reduce the time and cost associated with experimental methods, accelerating the drug discovery process.

2.3 Research Objectives

Objective 1: Harnessing Dataset Diversity

- Utilize DL approach for optimal dimensionality reduction of data metrics.
- develop a robust DL-based model to do unsupervised clustering on a dataset of small molecules at large scale.

Objective 2: CPI Model Development for Inductive Link Prediction

- Design and implement a ML model capable of accurately predicting CPI.

- Utilize the CPI model for inductive link prediction, contributing to the identification of novel drug candidates.

Chapter 3: Deep Clustering of Small Molecules at Large-Scale via Variational Autoencoder Embedding and K-means

3.1 Introduction

In the practice of drug discovery, high-throughput screening (HTS) is the primary approach for identifying drug candidates from chemical libraries (40). Nevertheless, screening is an expensive and time-consuming process, especially with the emergence of multidrug-resistant and extensively drug-resistant infections, which create formidable obstacles and challenges for this conventional drug discovery pipeline. To this end, various ML models have been developed and integrated as part of routine protocols in chemical and biological applications for decades (41). For instance, quantitative activity-structure relationships (QSAR) and quantitative property-structure relationships (QSPR) models have played a major role in molecular property predictions, one of the central tasks in the field of drug discovery (42–44). On the other hand, unsupervised ML methods have been extensively applied in the contexts of exploring molecular data sets and discovering the underlying molecular mechanisms of action (MOA) of new drugs (45). To establish an efficient ML model for chemical-related tasks, two core questions need to be answered: 1. how to encode a molecule in a machine-interpretable representation with the inclusion of informative and unique features of compounds (molecular featurization); 2. How to ensure the molecular database is diverse enough so that a ML model can learn sufficient chemical patterns to predict the desired properties outside of the training data.

In general, molecular representations can be divided into two main categories: calculated descriptors or fingerprints, and representations that are aggregated from molecular graphs (46). Chemical descriptors and fingerprints are deterministic characterizations of molecules in cheminformatics, and they are commonly employed as the input of conventional QSPR/QSAR models. For instance, ECFP, a type of topological fingerprints that characterize molecular structures through circular atom neighborhoods, are wildly adopted in QSPR/QSAR models (47). On the other hand, a molecular graph is a non-Euclidean structural representation composed by a set of atoms (V), and a set of chemical bonds or interactions (E) between each pair of adjacent atoms (48). In principle, the molecular graph can be treated as a connected undirected graph G defined by a set of nodes (V) and edges (E). In practice, various chemical properties can be calculated for each atom/bond (local features), so that a molecular graph is initialized by an atomic feature matrix (x_v) and a bond feature matrix (e_{vw}). To utilize local features of molecules for cheminformatics tasks such as molecule property prediction or clustering, the atomic and bond features need to be aggregated to the molecular level.

Clustering belongs to the unsupervised ML, which discovers the existing patterns in a given dataset and classifies the objects into similar groups (49). In bioinformatics, a variety of clustering algorithms have been implemented depending on different tasks and data (50,51). Before developing a QSPR/QSAR model, it is necessary to carry out the clustering analysis of compounds in the virtual chemical database for three reasons. First, as the quality of predictions from a data-driven model is largely determined by the dataset, validating the diversity of compounds in the selected virtual library ensures the model can learn sufficient chemical information and make

decent predictions. In addition, by identifying the similarity or heterogeneity among the chemicals contained in the data, a more comprehensive understanding of the underlying mechanism of action (MOA) of drugs could be gained. Furthermore, this may lead to a broader objective concerning the selection of compounds for the establishment of a dataset for chemical-based ML tasks, which is known to be a challenging and costly procedure (43). Knowing the categories of chemicals that need to be included in the dataset can greatly reduce the number of molecules that should be screened in the laboratory while ensuring the quality of the dataset for the model building at the same time.

In this study, we developed a novel molecular embedding learning approach that combines both PCA and a VAE to integrate the global and local features of molecules for clustering the ~50,000 chemicals collected in the large-scale chemical-genetic interaction profiles (CGIP) of mutant strains of the bacterium Mycobacterium tuberculosis (52). This work provides an in-depth analysis of the large-scale chemical library that was screened against *Mycobacterium tuberculosis* mutant strains (hypomorphs). Moreover, by investigating the generated compound clusters, we highlight the importance of feature engineering and gained insight into clusters of compounds that may target the same biological systems, and thus may possess similar biological functions.

3.2 Materials and Methods

3.2.1 Overall Study Design

The study framework included three parts: molecule featurization, clustering analysis and evaluation (**Figure. 3-1**). The framework started by the feature engineering of the compounds. To take better advantage of both the global and local features of molecules, chemical descriptors, atomic and bond features were first generated from RDKit (53). The atomic and bond feature matrices for each molecule were compressed by the principal component analysis (PCA) (54), then incorporated in the clustering analysis along with the chemical descriptors. With the composite representations of molecules, we selected the optimum number of clusters based on the analysis of Silhouette method. Next, using the obtained hyper-parameter, we investigated three clustering methods: K-means, K-means with autoencoder, and balanced iterative reducing and clustering using hierarchies (BIRCH). Lastly, we evaluated and compared the clustering methods on three internal indices, and visualized examples from five clusters by means of similarity maps.

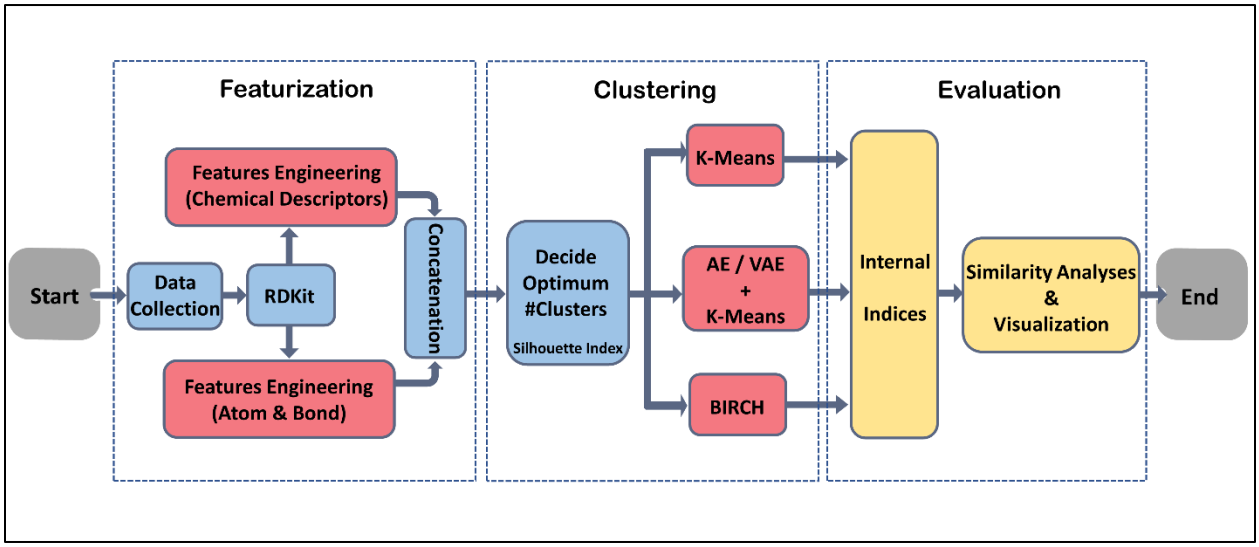

Figure 3-1 Schematic representation of the study design

3.2.2 Data Sources

The Johnson et al. (52)dataset used in this study has been made publicly available on the website (https://www.chemicalgenomicsoftb.com), where they provided the structure and function annotation of 47,217 compounds represented in the SMILES (13). These compounds were collected and screened against 153 Mycobacterium tuberculosis strains (strain H37Rv and hypomorphs) to identify potential antimicrobial drugs that may possess growth-inhibitory to wild type Mycobacterium tuberculosis. The relative abundance of each unique strain was calculated after drug exposure, then the number was compared to the untreated dimethylsulfoxide (DMSO) control by maximum likelihood estimation of the natural fold change (LFC). LFC estimates mark the differences between the experimental and control groups, where larger absolute values reveal the higher antibiotic activities of drugs against specific *Mycobacterium tuberculosis* strain. Focusing on the analysis of the distribution and diversity of chemicals in the Johnson et al. (52) library, only the 47,217 SMILES strings, which describe the structural information of molecules, were used in this study.

3.2.3 Feature Engineering of Molecules

Feature engineering (**Figure 3-2**) is the first component of our framework, which transforms the SMILES notation of each compound to a numerical format that serves as input to the clustering analysis that follows. To depict compounds with both chemical properties and structural relationships, in general, we fused the global and local features together by performing a series of concatenations and dimensionality reductions.

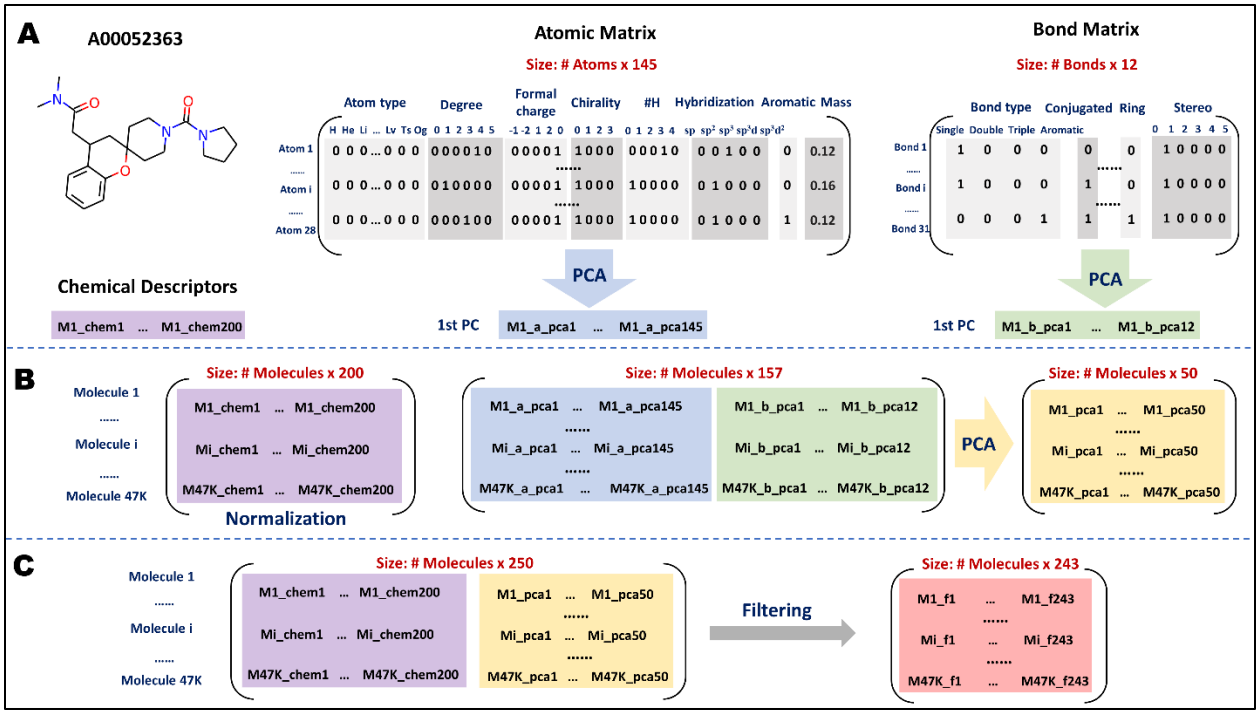

Figure 3-2 Overview of the feature engineering. (A) The first compound of the Johnson et al dataset (compound identifier: A00052363) is used as an example. Firstly, 200 chemical descriptors were derived from RDKit. On the atomic/bond level, 8 types of atomic features and 4 types of bond feature were generated and one-hot encoded. Next, PCA was performed on each transposed atomic and bond matrix.

The first principal component (PC), which contains the greatest amount of variance, was selected as the one-dimensional representation for each feature matrix. (B) The same process was used to iterate through all the compounds in the dataset. We concatenated the atom and bond features and selected the top 50 features which explained all the variance by performing another PCA on the matrix. For molecular descriptors, we normalized the values with Z-score scaling among samples. (C) For each molecule, we concatenated the normalized chemical descriptors with its' aggregated local features. Finally, we filtered out columns with zero variance, resulting in a feature matrix of size (47217×243) for the subsequent clustering.

3.2.4 Calculation of Molecular Descriptors

A collection of 200 descriptors were derived from different modules in RDKit package (53) (**Figure 3-2A -left panel**), ranging from basic descriptors such as molecular weight and the number of radical electrons, to topochemical descriptors (e.g. Balaban's J index) and hybrid Estate combining VSA descriptors (e.g. MOE VSA descriptors), etc. (55). The comprehensive cheminformatics descriptors include a wide range of chemical properties at the molecular level, providing a rich source of chemical information on a variety of aspects.

3.2.5 Generation of Atomic and Bond Feature

As defined in the introduction, a molecular graph consists of an atomic matrix (x_v) and a bond matrix (e_{vw}). **Table 3-1** shows the 8 types of atomic features and 4 types of bond feature used in this study. All atomic and bond features were one-hot encoded (**Figure 3-2A -middle and right panels**), except for the atomic mass, which was scaled by dividing by 100. Encoding features in a one-hot manner is a common technique in categorical data, which guarantees the algorithm doesn't consider higher numbers to be more important and allows for a more expressive representation of categorical data (56).

Table 3-1. Descriptions of Atomic and Bond Features

Feature Type	Attribute	Size	Description
	Atom type	118	known chemical elements (by atomic number)
	Degree	6	number of bonds the atom is involved in
	Formal charge	5	electronic charge assigned to an atom
	Chirality	4	unspecified, tetrahedral CW/CCW, or other

			types of chirality
Atomic Features	Number of H	5	number of bonded hydrogen atoms
	Hybridization	5	sp, sp^2, sp^3, sp^3d, or sp^3d^2
	Aromaticity	1	whether the atom is aromatic
	Atomic mass	1	mass of the atom
Bond Features	Bond type	4	single, double, triple, or aromatic
	Conjugated	1	whether the bond is conjugated
	Ring	1	whether the bond is in a ring
	Stereo	6	stereochemistry of bonds (none, any, E/Z or cis/trans)

3.2.6 Feature Aggregation

To utilize graph representations of a molecule, the features of atoms and bonds need to be aggregated, then embedded into a vector (read out) for use in subsequent tasks. In this regard, many GNN have been proposed, in which molecular features were aggregated via different message passing (or graph convolution) scheme (44,57,58). However, GNNs belong to supervised algorithms, where the ground truth for each molecule is required during training. In other words, the local messages of molecule can only be updated iteratively via backpropagation on the gradients of the loss between current states and targets. Since we only make use of the SMILES strings in the library and do not have the ground truth for clustering, we propose a PCA-based approach to the aggregation of molecular local information.

PCA is an unsupervised technique of dimensionality reduction that works by finding a new set of mutually uncorrelated variables (principal components) to represent the original data while retaining most of the variation (54,59). In this study, we used the linear PCA, which projects the data onto linear subspaces, for the purpose of aggregating the local features to a lower dimension. Specifically, we performed a linear PCA on each transposed molecule-specific atomic and bond matrix, respectively. The first principal component, which contains the greatest amount of variance, was chosen as the one-dimensional representation of each atomic and bond feature matrix of a given molecule, respectively (**Figure 3-2A -middle and right panels**). In this way, the local features of different sizes in each molecule were aggregated into a representation with the same dimensionality for all molecules (**Figure 3-2B -middle panel**).

To further filter out the redundant features with low variance across the molecules, we performed another PCA on the concatenated atomic and bond feature matrix (**Figure 3-2B -middle panel**) and selected the top PCAs or features (**Figure 3-2B - right panel**) which explained all of the variance (**Figure 3-3**). For the molecular descriptors, we performed a Z-score normalization so that the values all fell within the same range, with the aim of preventing features with larger absolute values from dominating the algorithms. Lastly, we concatenated the resulting local and global features, followed by a filtering operation which delete the feature columns with zero variance (**Figure 3-2C**). The final representation of a molecule is in size of 243, which incorporate abundant local and global information for the subsequent clustering of the molecules (**Figure 3-4**).

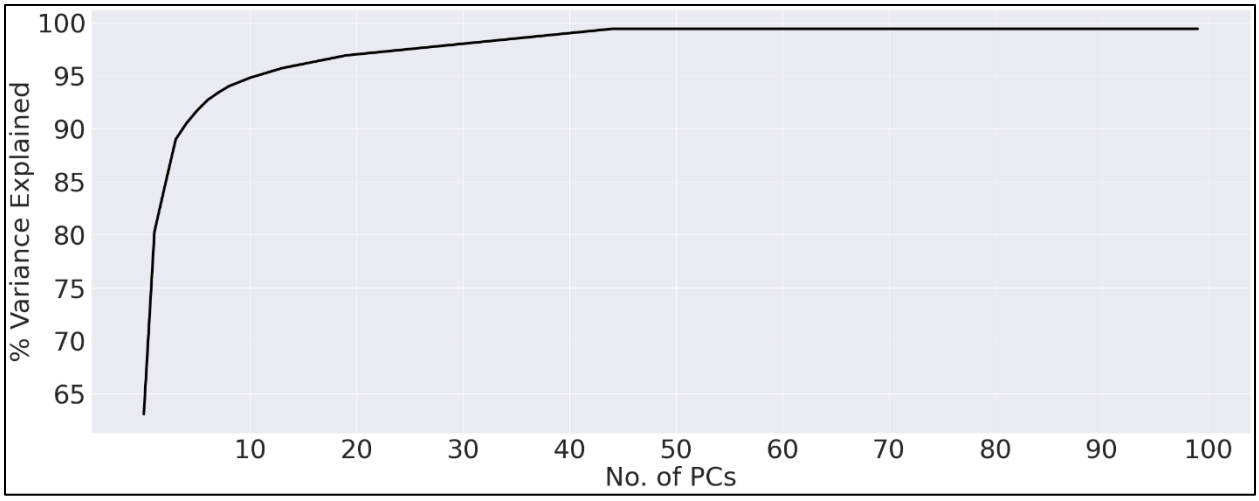

Figure 3-3 The proportion of variance explained based on the number of PCs. The first 50 PCs explained 100% of the variance of the data.

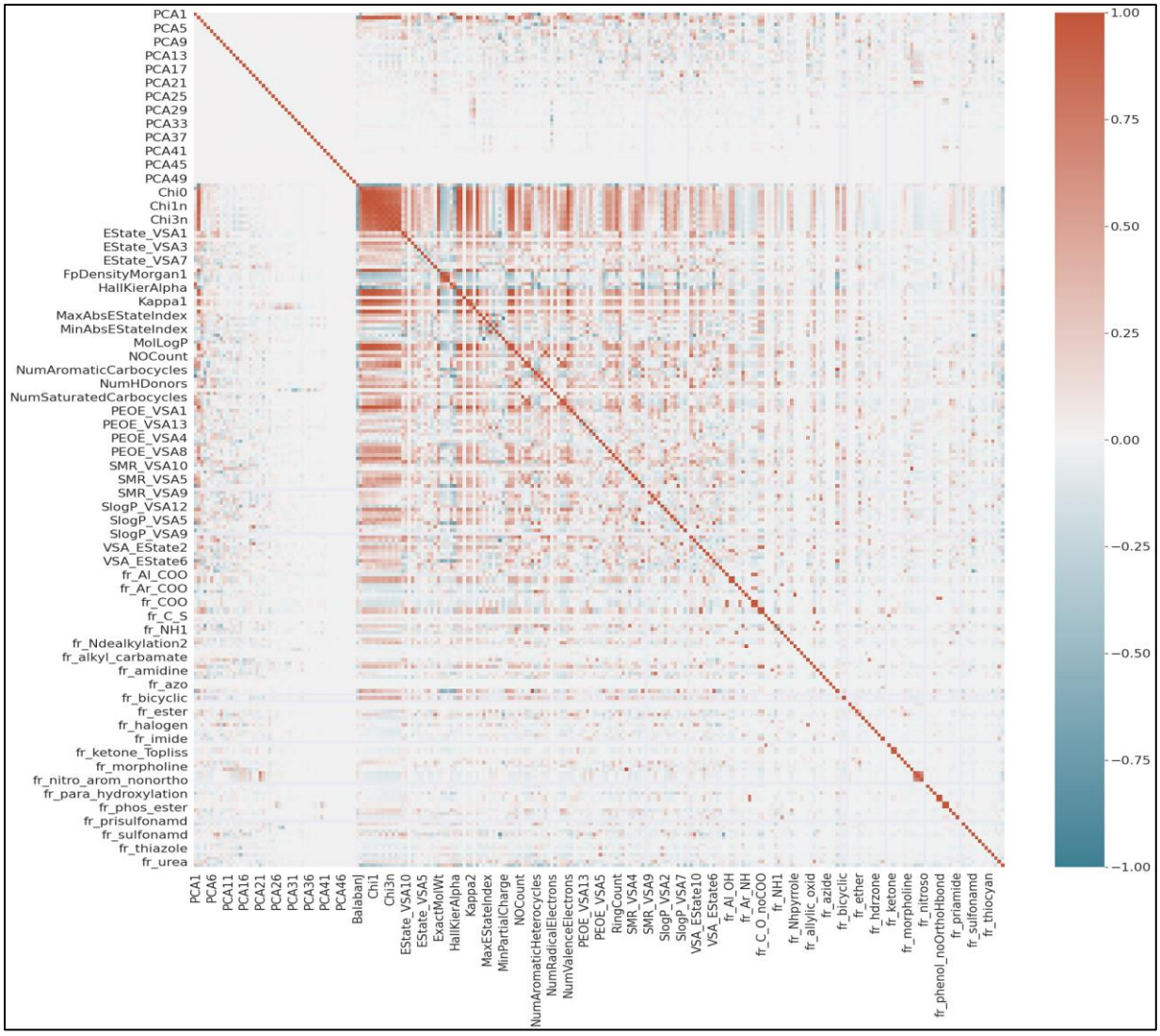

Figure 3-4 Heatmap showing the correlations between each pair of the features. The 50 aggregated atomic and bond features are named as PCA (1-50), while the rest are the names of the molecular descriptors generated by RDKit.

3.2.7 Molecule Clustering

Due to the small molecules at large-scale, we selected below four clustering methods since they can be scalable for very large datasets, perform data reduction, and be efficient in memory and time usage.

3.2.8 K-Means Method

K-means (60) is one of the simplest and most famous clustering algorithms. K-means starts to indicate centroids (a centroid is the center of a cluster of molecules) randomly. A molecule is in a particular cluster if it is closer to that cluster's centroid than any other centroid. K-means tries to finds the best centroids by alternating between assigning molecules to clusters based on the current centroids and choosing centroids based on the current clusters of the molecules until it finds a convergence. It stops creating and optimizing clusters when either the centroids have been stabilized or the defined number of iterations has been achieved.

3.2.9 BIRCH Method

BIRCH (Balanced iterative reducing and clustering using hierarchies) is an unsupervised data mining algorithm used to perform hierarchical clustering over particularly large data sets by first generating a small and compact summary of the large dataset that retains as much information as possible (61). Hence, each clustering decision is made locally without scanning all molecules and currently existing clusters. The method makes full use of available memory to derive the finest possible sub-clusters while minimizing the input/output costs.

3.2.10 DL Autoencoder-based K-means Clustering

An autoencoder (AE) is a type of neural network that transforms input molecule-specific features into its output (62). An autoencoder consists of two parts in this transformation:

1. The Encoder transforms the high dimensional inputs into a smaller set of dimensions while keeping the most important latent features.

2. The Decoder that uses the reduced set of latent features to reconstruct the initial input data.

The autoencoder algorithm makes embedding of the large molecule-specific feature data and reconstructs it in a lower dimension without losing important information. We used K-means to cluster molecule-specific embeddings and generate molecule clusters, which is expected to have much better performance and capture the cluster` labels (63).

3.2.11 DL Variational Autoencoder-Based K-means Clustering

Although AE is simple, it is hard for us to control over how we model our latent distribution. A variational autoencoder (VAE) (64) is a type of generative neural network based on an autoencoder that is made from an encoder and a decoder. VAE makes embedding of the input molecule-specific features to a latent space in a probabilistic manner and reconstructs the input data from the latent space. Hence, VAE makes it more practical and feasible for large-scale data sets, like the set of molecules we analyzed here.

The general architecture of the VAE algorithm is summarized in **Figure 3-5**. The goal is to minimize the VAE loss that defines as follow,

Reconstruction Loss:

$$L = \frac{1}{m}\sum_{j=1}^{m} l(x^j, \hat{x}^j) \quad , (3\text{-}1)$$

where m is the number of molecules, x is the input, and $\hat{x}$ is the output.

VAE Loss:

$$L(x, \hat{x}) = l_{reconstruction} + \frac{\beta}{2}\sum_{i=1}^{d}(V(Z) - log\,[V(Z)]\; - 1 + E(Z)^2)_i \text{ , (3-2)}$$

where x is the input data, $\hat{x}$ is the output data, β is the hyperparameter, V (Z) is the variance of the inputs in the encoder section, and E (Z) is the mean of the molecules in the encoder section.

In the VAE model used with our framework, the encoder accepts samples of molecule`s features data from the input. The encoder contains the combination of six layers of linear and batch normalization layers and an output layer that produces embeddings with reduced dimension of the samples described above. The decoder subnetwork accepts these encoded samples as input, passing these through an architecture like the encoder, which reconstructs the original samples. In both subnetworks, the activation function of the hidden layers is a rectified linear unit ReLu(.). An Adam optimizer with a learning rate of 1e-3 was used to update the neural networks' weights.

Using the embeddings from the molecule-specific features based on VAE, we could apply the K-means algorithm to generate the molecule clusters based on the predefined number of clusters.

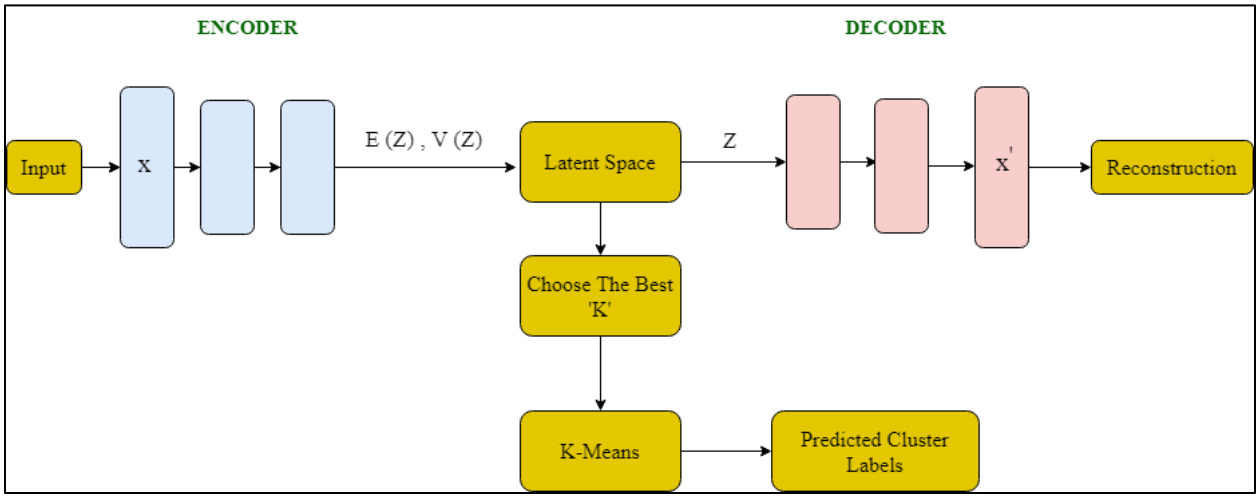

Figure 3-5 Schematic of the Variational Autoencoder model + K-Means. Encoder: X→ R2d , Decoder: Z→ Rn . E(Z) represents the mean of the points, and V(Z) is the variance of the points.

3.2.12 Estimation of the Number of Molecule Clusters

One of major challenges in performing clustering analysis is to decide the number of clusters in a given input data. One of the most popular methods to calculate this number is the Silhouette index (65). Silhouette index is a measure for the validation and interpretation of the consistency within data clusters. This approach gives a simple graphical representation of how well each object has been grouped. The Silhouette coefficient s is expressed as:

$$s = \frac{(b-a)}{max\,(a,b)} \quad , (3\text{-}3)$$

where a is the mean distance between a given molecule and all other molecules in the same cluster while b is the mean distance between a given molecule and all other molecules in the next nearest cluster. Silhouette coefficient values range between −1 and +1, with higher values indicating that the molecules are better clustered.

3.2.13 Clustering Performance Evaluation

It is a crucial part to measure the quality of a clustering algorithm so that we can choose the clustering algorithm that performs best for an input set of large-scale molecules. Generally, there are external and internal evaluation measures. External evaluation measures usually require a ground truth which is not available in our study. Hence, we focused on the internal clustering validation. In particular, we applied three widely used performance measures, the Silhouette coefficient, the Calinski-Harabasz index (66) and the Davies-Bouldin index (67), to evaluate our clustering performance. The internal clustering measurements were implemented with the "sklearn" python package (68).

3.2.14 Calinski-Harabasz Index

The Calinski-Harabasz index is a method for finding the ratio of the sum of between-clusters dispersion and of inter-clusters dispersion of all clusters identified from the analysis (66). The Calinski-Harabasz score *S* for *k* clusters is given as:

$$S = \frac{tr(B_k)}{tr(W_k)} \times \frac{n_E - k}{k-1}, \quad (3\text{-}4)$$

where the $tr(B_k)$ is the trace of the between-group dispersion matrix and $tr(W_k)$ is the trace of the within-cluster dispersion matrix defined by:

$$B_k = \sum_{q=1}^{k} n_q (c_q - c_E)(c_q - c_E)^T, \quad (3\text{-}5)$$

$$W_k = \sum_{q=1}^{k} \sum_{x \in C_q} (x - c_q)(x - c_q)^T, \quad (3\text{-}6)$$

where C_q is the set of molecules in the cluster q. c_q is the center of the cluster q. n_q is the number of molecules in the cluster q. c_E is the center of the cluster E. A higher score indicates a model with more separated clusters.

3.2.15 Davies-Bouldin Index

The Davies–Bouldin index is a measure to evaluate cluster analysis algorithms (67). This metric is an internal evaluation scheme, which is defined as the similarity of the average between each cluster Cu, for u = 1, … , k, and its most similar one Cv . Ruv is a measure that defines the similarity given by:

$$R_{uv} = \frac{s_u + s_v}{d_{uv}}, \quad (3\text{-}7)$$

where s_w is the diameter of a cluster for w = 1, …, k and duv is the distance between cluster centroids u and v. The Davies-Bouldin (DB) index can be expressed as:

$$DB = \frac{1}{k} \sum_{u,v=1}^{k} max R_{uv}, \quad (3\text{-}8)$$

where a lower Davies-Bouldin index indicates a model with better separation between the clusters.

3.2.16 Visualization Analysis

3.2.16.1 t-SNE Visualization of the Molecular Embeddings

t-SNE is a statistical tool to visualize high-dimensional data by giving each data point a location in a two or three-dimensional map in such a way that similar objects are modeled by nearby points and dissimilar objects are modeled by distant points with high probability (69,70). We applied the t-SNE to visualize our embeddings from the VAE analysis.

3.2.16.2 Molecular Similarity Map

In cheminformatics, a common strategy to quantify the similarity between two compounds is by assessing the fingerprint similarities with distance metrics, such as Dice (71) or Tanimoto (72). Based on this scheme, the similarity map proposed by Riniker et al. (73) provides the ability to visualize the atomic contribution to the similarity between two molecules, or the predicted probability from a given ML model. For each atom in a test compound, its atomic contribution (weight) to the similarity to a reference compound equals to the similarity difference when the bits in the fingerprint corresponding to the atom are removed. The weights generated for each atom are then normalized and used to color the topography-like map for visualization. We generated the molecular similarity map using the module implemented in the RDKit.

3.3 Results

3.3.1 Estimating the Number of Clusters

Using a range of 5 to 200 clusters with a step size of 5 and different numbers of embeddings (16, 32 and 64) from the AE and VAE algorithms, we applied the Silhouette method to estimate the Silhouette coefficient values (**Figure 3-6**). As shown in **Figure 3-6**, all the feature sets or embeddings achieve relatively stable Silhouette coefficient values at cluster size 50. Using the 243 raw features produced the lowest Silhouette coefficient value while the best embeddings are the 32 latent features from the VAE algorithm with a Silhouette coefficient value 0.286 at the cluster size 50 (**Figure 3-6).**

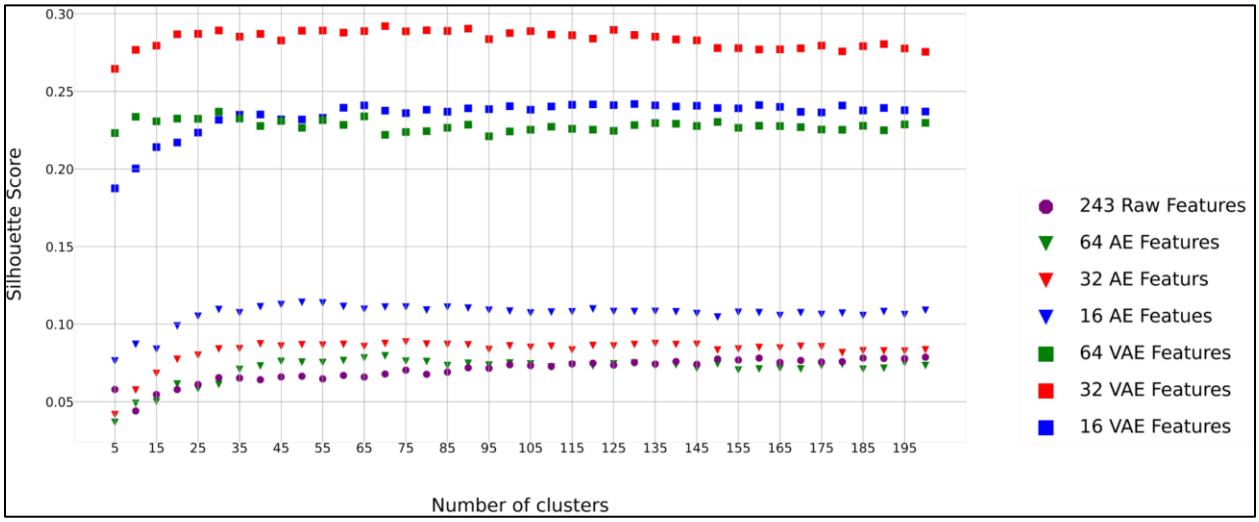

Figure 3-6 Embedding-specific silhouette scores under different number of molecule clusters. Results based on the 243 raw features and the 64, 32, and 16 embeddings from the VAE and AE algorithms, respectively.

3.3.2 Performance Evaluation of the Identified Molecule Clusters

Table 3-2 summarizes and compares the clustering performance of the four suggested algorithms (K-means, BIRCH, AE + K-means and VAE + K-means) based on the 243 raw features and their embeddings of the molecule data set from VE and VAE, respectively. For the K-means and BIRCH, we determined the optimal number of clusters 30 based on the 243 raw features (**Figure 3-6**). For AE + K-means and VAE + K-means, we determined the optimal number of clusters based on different number of embeddings (16,32, and 64) (**Figure 3-6**, **Table 3-2**). Overall, based on the three internal measurement indexes, we found the algorithm of VAE + K-means with 32 embeddings showed the best performance (Calinski-Harabasz Index: 10112.928, Silhouette Index: 0.286, and Davies-Bouldin Index: 0.999) with 50 optimized clusters while K-means and BIRCH with the 243 raw features showed the worst performance (**Table 3-2**).

Table 3-2. Internal measurement indexes

Clustering Method	**#Clusters**	**Internal Indices**		
		Calinski-Harabasz	Silhouette	Davies-Bouldin
K-means	30	1010.383	0.066	2.167
BIRCH	30	825.288	0.042	1.964
VAE (16) + K-means	50	5545.491	0.236	1.142
VAE (32) + K-means	**50**	**10112.928**	**0.286**	**0.999**
VAE (64) + K-means	70	4965.177	0.229	1.183
AE (16) + K-means	50	1498.595	0.116	1.703
AE (32) + K-means	40	1117.688	0.085	1.912
AE (64) + K-means	70	717.636	0.075	2.260

3.3.3 Visualization of the Identified Clusters

We evaluated the distribution of the molecules in each cluster based on the number of molecules in each cluster using the results from the VAE-based K-means clustering with 32 embeddings (VAE (32) + K-Means) and 50 clusters (**Figure 3-7**). As shown in **Figure 3-7**, more than 80% of the clusters with more than 500 molecules and the cluster size is relatively homogeneous.

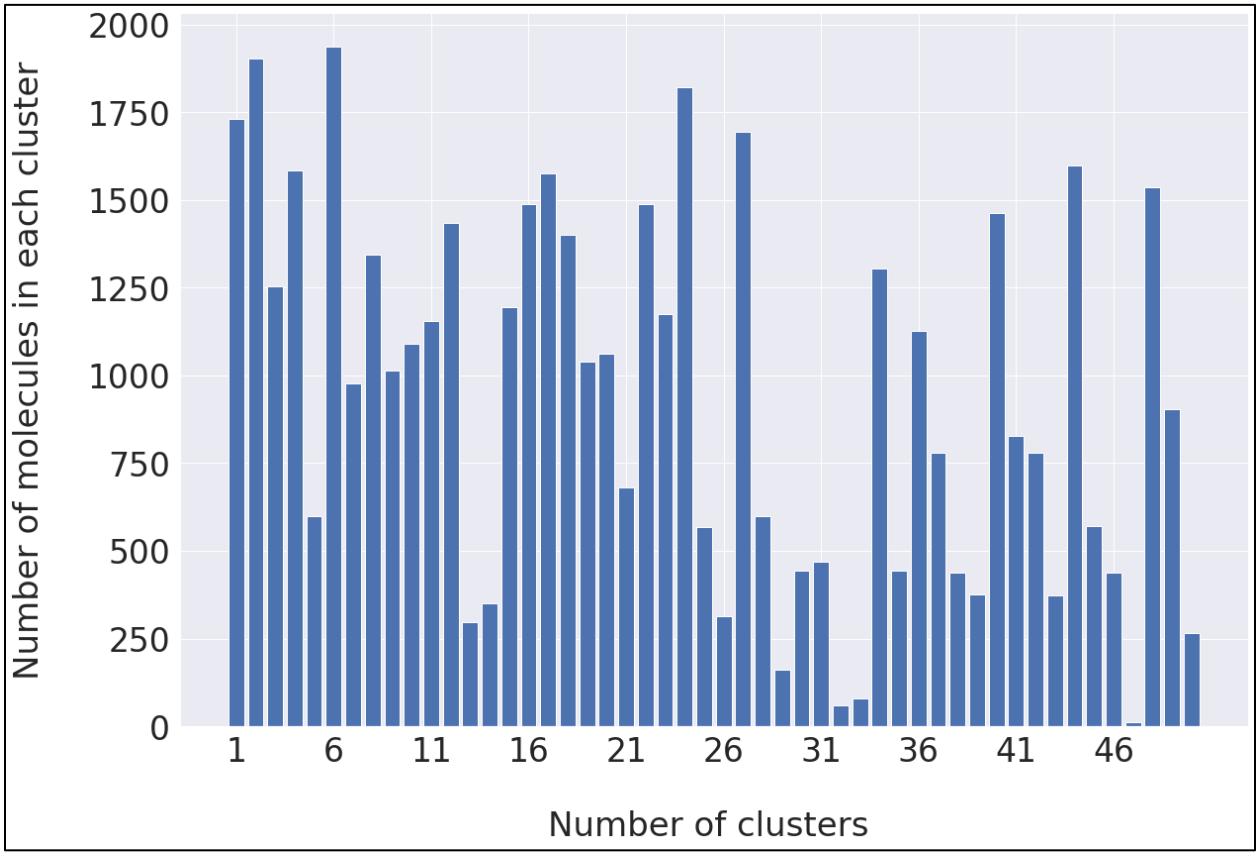

Figure 3-7 Distribution of molecules in each cluster

Furthermore, we visualized the embeddings from the results with the best algorithm (VAE (32) + K-means) using the t-SNE method (**Figure 3-8)**. Overall, the clustered molecules using the VAE (32) + K-means with 50 clusters showed consistent patters with the t-SNE analysis of the embeddings, that is, the t-SNE clustered majority of the cluster-specific molecules from the VAE (32) + K-means together.

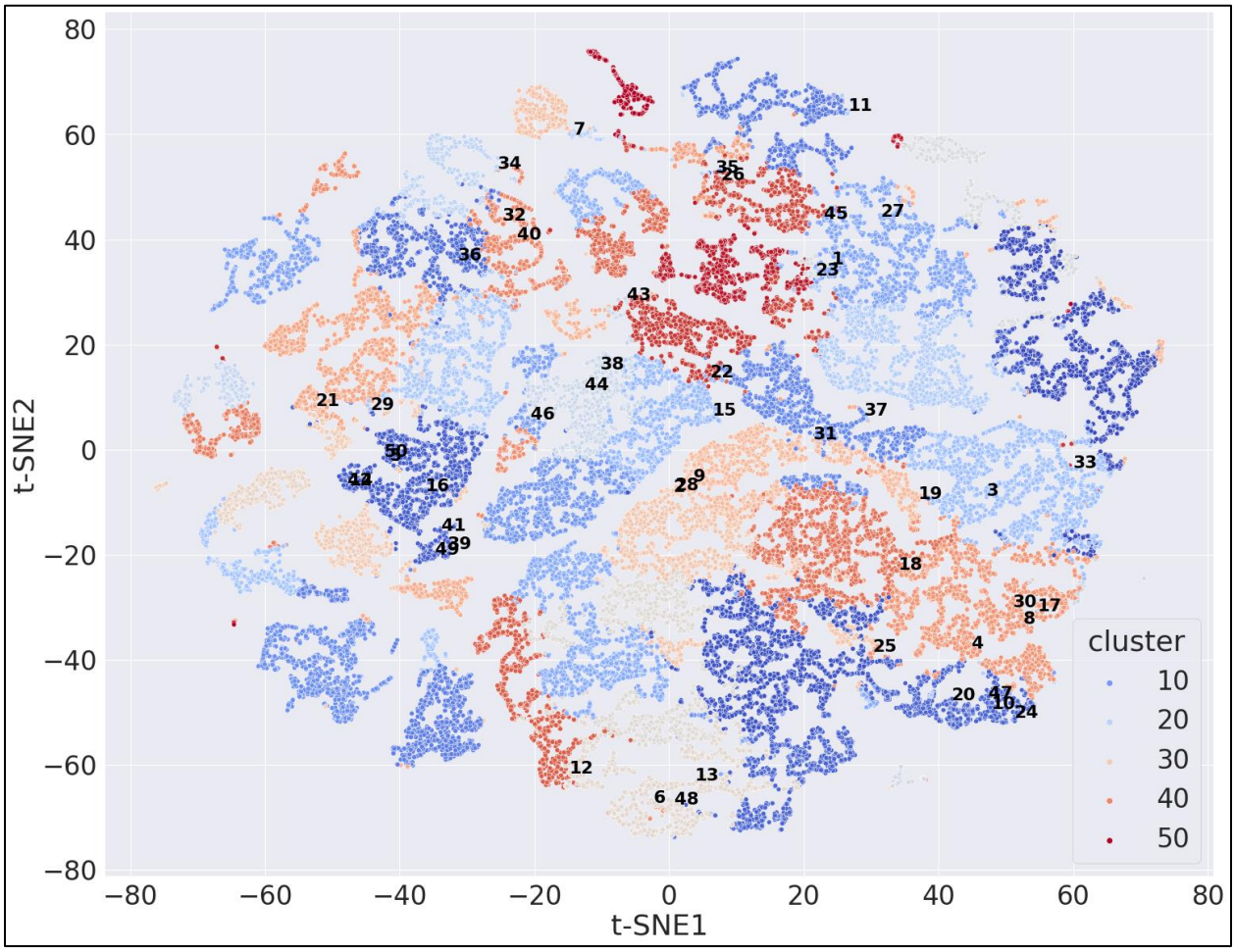

Figure 3-8 The t-SNE visualization of the 32 embeddings from VAE algorithm. The numbers are the cluster IDs from the results of VAE (32) + K-means. The colors represent the t-SNE analysis results.

To further examine the effectiveness of our clustering framework and discover the commonalities in molecular structures within the same cluster, four samples, including one reference molecule and three test molecules, were randomly selected from each of five randomly selected clusters and visualized (**Figure 3-9**). During the generation of the similarity maps, the count based ECFP with radius 2 and 2048 bits was used as the compound representation. In addition, the Tanimoto was selected as the metric during the fingerprint comparation as it is one of the best choices for fingerprint-based similarity calculation reported by Bajusz et al. (74) . In the similarity maps,

atoms contribute to the similarity score between the reference compound and the test compound are highlighted in green, whereas the red represents the opposite contribution.

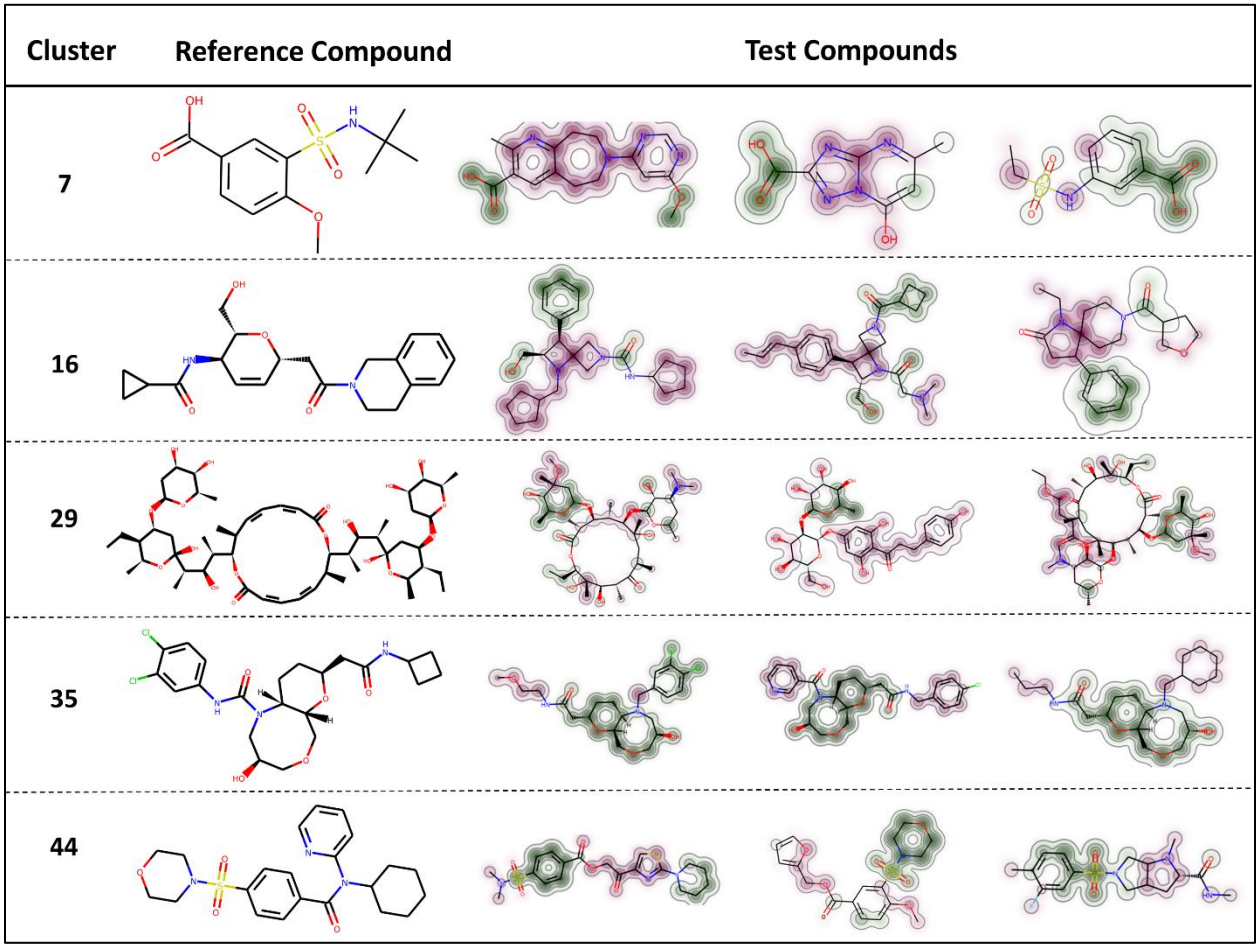

Figure 3-9 2D structure and similarity map for the examples randomly selected in the five clusters. For each cluster, the similarity scores between the reference compound and three test compounds were measured by the Tanimoto metric using the count based ECFP (radius=2, bit=2048). The similarity weights were visualized by colors on the structure (similarity maps). Sub-structures that increase the similarity score were presented in green, whereas the red indicates the opposite.

From the randomly selected cases in **Figure 3-9**, our clustering framework successfully grouped molecules with more structural similarities into clusters. For instance, all four molecules in cluster 7 contain aromatic carboxylates and they were labelled in green, indicating the similarity. Aryl

halides appear in three samples in cluster 35, and all samples from cluster 44 contain sulfonamides. We also show the pairwise similarities scores between all selected samples in one matrix (**Figure 3-10**) to present how samples differ within clusters. In order to generate a matrix with a larger contrast, we chose binary ECFP (radius=1, bit=2048) as the molecular representation and calculated the Tanimoto score between them. The matrix is diagonally symmetric and orange rectangles denote samples that belong to the same cluster. The more similar two molecules are, the greater the value of Tanimoto between them. As shown in **Figure 3-10**, it is clear that samples originating from the same cluster obtained larger Tanimoto scores and exhibited in darker colors in the matrix. Cluster 35, in particular, has a distinctive difference in color from samples not in this cluster. Mol1 and Mol3 in cluster 29 achieved the highest similarity score (0.86). From their structure in **Figure 3-9**, we can also identify the characteristics of structural closeness between them.

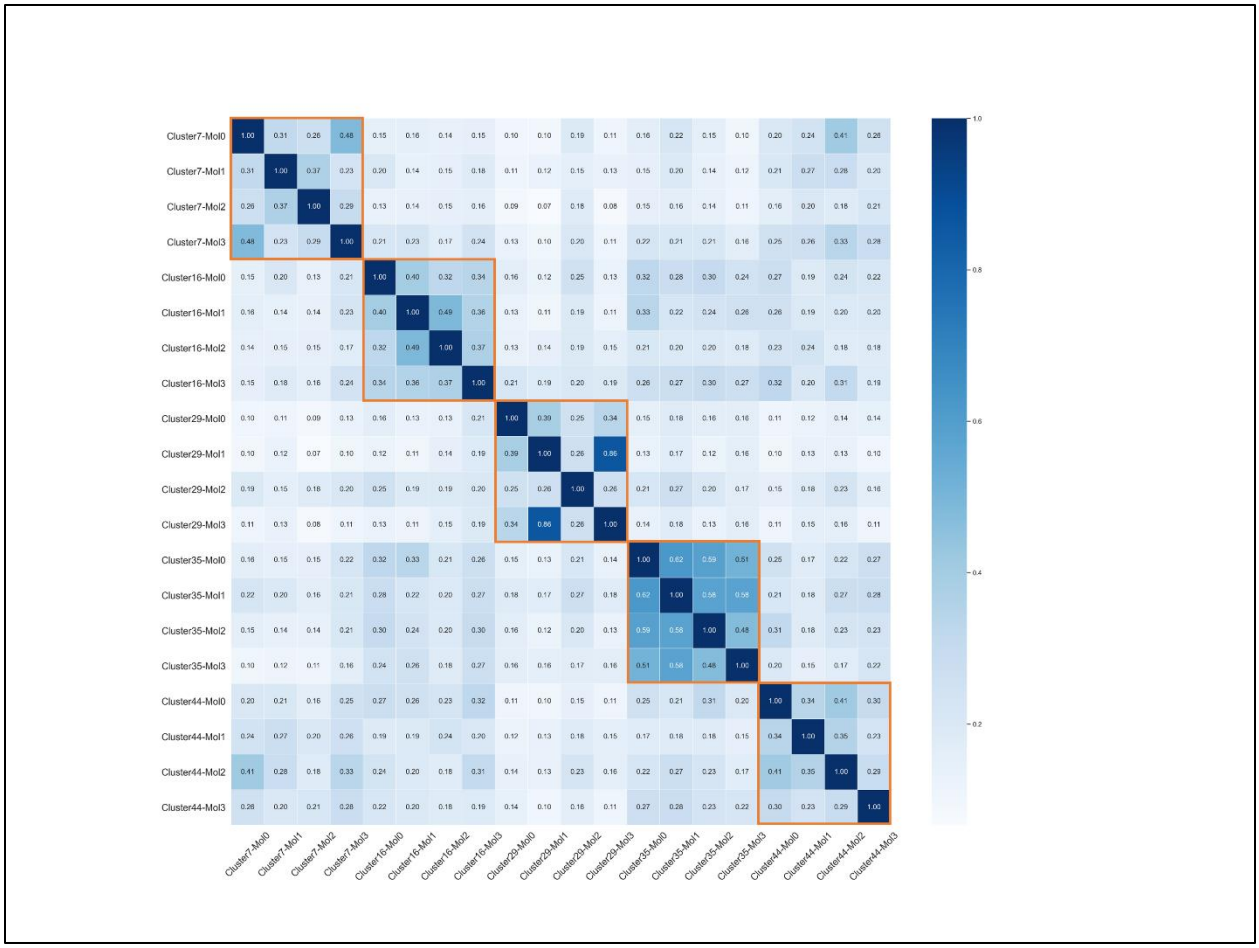

Figure 3-10 Tanimoto similarity matrix between each pair of the examples. It includes the reference compounds (Mol0) and three test compounds (Mol1, Mol2, Mol3). The binary ECFP (radius=1, bit=2048) were used for the similarity calculation. The orange rectangle circles the samples belonging to the same cluster.

3.4 Discussion

In this study, we first tried to capture molecular descriptors, atomic features and bond features. However, for a given molecule, the molecular descriptor is a feature vector while the atomic features and bond features are two different matrixes with different dimensions. We explored a simple PCA method to reduce the atomic feature matrix and the bond feature matrix to a PCA-based feature vector, respectively. Since we only focused on the first PC in each of the two feature matrices, this may lose some information in the data.

To further integrate the molecular descriptors, PCA-based atomic features and PCA-based bond features, we made the embeddings with different sizes using the standard AE and VAE algorithms, respectively. We used a simple K-means clustering method to evaluate the performance of the embeddings in clustering the large-scale molecules. Overall, we showed that VAE-based embeddings have significant better performance than other embeddings to cluster the molecules. When applying K-means for clustering analysis, one major challenge is to predefine the number of clusters in the data. Here, we applied the widely used Silhouette method to estimate the number of clusters in the large-scale molecule set. However, we expect some other soft K-means methods (75) can achieve similar performance to the methods we applied here.

Comparing with the clustering applications in other domains, such as disease subtyping using gene expression profiles, we found that the molecular cluster separation score measured by the Calinski-Harabasz index is relatively low (the maximum one is 0.286). We also found the normalization strategy used for the molecular descriptors can have a significant impact on the clustering results. All these suggest that we may need to extract more molecule-specific features, such as three-dimensional coordinates of the molecular structure, establish more robust preprocessing pipeline, and make new embedding strategies to perform deep clustering of the molecules.

3.5 Summary

In this study, we developed a novel molecular embedding framework that combines both PCA and VAE to integrate the local and global features of molecules. To evaluate the usefulness of the molecular embeddings, we applied our methods to extract the embeddings of the ~47,000 molecules from a large-scale molecule library that were collected and screened against *Mycobacterium tuberculosis* strains and performed an in-depth clustering analysis of the embeddings by comparing a variety of unsupervised clustering algorithms, including standard K-means, K-means with AE, K-means with VAE, and BIRCH. We demonstrated that embeddings of the molecules using the VAE-based method have significant advantages over those based on the AE-based method. However, our model cannot produce an index shows the diversity of a chemical dataset and use it to compare the diversity of two different datasets, but it can be applied to any large-scale chemical libraries to determine if it is diverse enough or not based on the needs of the projects. Also, our novel analytical framework based on the clustering analysis, may provide insights for optimizing drug discovery by decreasing the size of screening libraries.

Chapter 4 GraphBAN: Inductive Graph-Based Prediction of Compound-Protein Interactions

4.1 Introduction

One of the initial steps in drug discovery is the identification of chemical compounds that bind a molecular target involved in a disease mechanism (76). In the case of protein targets, visualizing the intermolecular forces that mediate CPIs is crucial for further improvements of the drug under investigation (77). One way to identify CPIs is by chemogenomic profiling, which systematically analyzes potential biological targets that interact with extensive collections of chemical compounds. Unfortunately, chemogenomic profiling of potential drugs can be lengthy, complex, and costly (78).

Therefore, computational methods that predict drug-target interactions (DTIs) can have a significant impact in terms of reducing the resources and time needed to discover and develop a new drug (79). Computational prediction of CPIs encompasses two primary approaches: structure-based and ligand-based methods (80). Structure-based methods require information on the interactions of the ligand with the protein in the site of interest. This necessitates the utilization of molecular docking simulations, which, regrettably, may not be universally applicable for all CPIs. In contrast, ML and DL techniques offer an in-silico alternative for analyzing CPI, effectively alleviating the burdens of time and cost by filtering out thousands of extraneous compounds in virtual databases.

Current methods aimed to predict CPIs or DTIs can analyze data from three different aspects: the first aspect is the input data, which includes two principal categories: graph-based and non-graph-based input data. The graph-based data generates a heterogeneous graph, where compounds, proteins or other biological entities constitute the nodes, while interactions form the edges. These graph structures are subsequently analyzed using GNN-based models to capture structural information of nodes and their associations. One advantage of this approach is that considering neighbouring features can improve link prediction accuracy. However, a disadvantage is that the training procedure is more complex and struggles with fully unseen nodes; having only node attributes is not an easy task. Notable examples are listed in references (39,81,82). On the other hand, the input data received as a non-graph CPI dataset is preferable due to its flexibility in tabular data analysis with several ML/DL models available and more efficient computational costs that lead to overall better results (83–86).

The second aspect pertains to how the trained ML/DL models are evaluated. There are two main approaches to do this: the first is whether the testing is performed on compounds and proteins that the model has encountered during training (transductive analysis) or on entirely different, previously unseen entities (inductive analysis). The third aspect involves assessing the similarity of the compounds and proteins in the test set to those in the training set based on their descriptors (take the descriptors of compounds/proteins as their domain). If they are similar, the analysis is referred to as in-domain analysis, whereas if they differ, it is classified as cross-domain analysis. In real-life scenarios to use the link predictor model, we will have inductive compounds/proteins that were not in our train set plus that with a highly chance they different in domain compared to our train set, so it will be more relevant to address our test scenarios in this way compared with having transductive and in-domain test sets. Abbasi et al. (87) introduced a model based on LSTM

and CNNs to do CPI prediction considering cross-domain analysis. Moreover, Kao et al. (88) ensembled different DL models to do DTI prediction and applied their test analysis on cross-domain data compared with their training data.

An unexplored path in CPI/DTI prediction involves utilizing graph input data to enable cross-domain inductive analysis, employing only the node attributes of the test sets while simultaneously benefiting from the advantages of treating the CPI train set as tabular data (non-graph approach). In this work, we introduce a unified model called GraphBAN, which includes the full spectrum of CPI prediction requirements, covering transductive, inductive, in-domain, and cross-domain analyses. Our proposed methods systematically tackle the three aspects of analyzing the CPI prediction mentioned above. First, we devised the graph autoencoder to attention (GAE2A), which combines GNN and attention mechanisms to handle link prediction in both transductive and inductive scenarios adeptly. Then, we incorporated KD to create a teacher-student model, referred to as the graph autoencoder to feature (GAE2F), enhancing the analysis of the nodes' features and achieving improved predictions in inductive scenarios. Finally, GraphBAN presents a comprehensive end-to-end model that replaces the BAN module with a simple node feature concatenation and a cross-domain adoption module, further elevating its generalization ability.

4.2 Methodology

We employed three distinct methodologies, each offering a unique perspective, to address the challenge of predicting CPIs. Our primary objective centers around leveraging graph topology to acquire an understanding of graph structural characteristics through the utilization of GNNs. Subsequently, we applied the trained model to both transductive and fully inductive nodes, employing a cross-domain approach with the ultimate aim of ensuring the model's applicability to real-world scenarios.

4.2.1 Method 1: GAE2A

The comprehensive architecture of this model is depicted in **Figure 4-1**, which draws inspiration from research conducted on inductive link prediction in RNA-Protein interactions utilizing GNNs (89). This approach entails the design of a model capable of handling both inductive and transductive link prediction scenarios. The model comprises two primary components: **Train block** is dedicated to training, with an input graph with the nodes` features, enabling transductive link prediction. **Unseen test block** shows an architecture for inductive analysis, employing an attention mechanism. In the training process, we employ a Graph Autoencoder (GAE) model to generate embeddings for compounds and proteins. These embeddings serve as input to the Deep Neural Network (DNN) block, facilitating the transductive link prediction. Additionally, we utilize the embeddings generated by the transductive block in the subsequent phase to produce embeddings for previously unseen compounds and proteins, leveraging an attention mechanism. The subsequent sections provide a detailed explanation of each of these blocks.

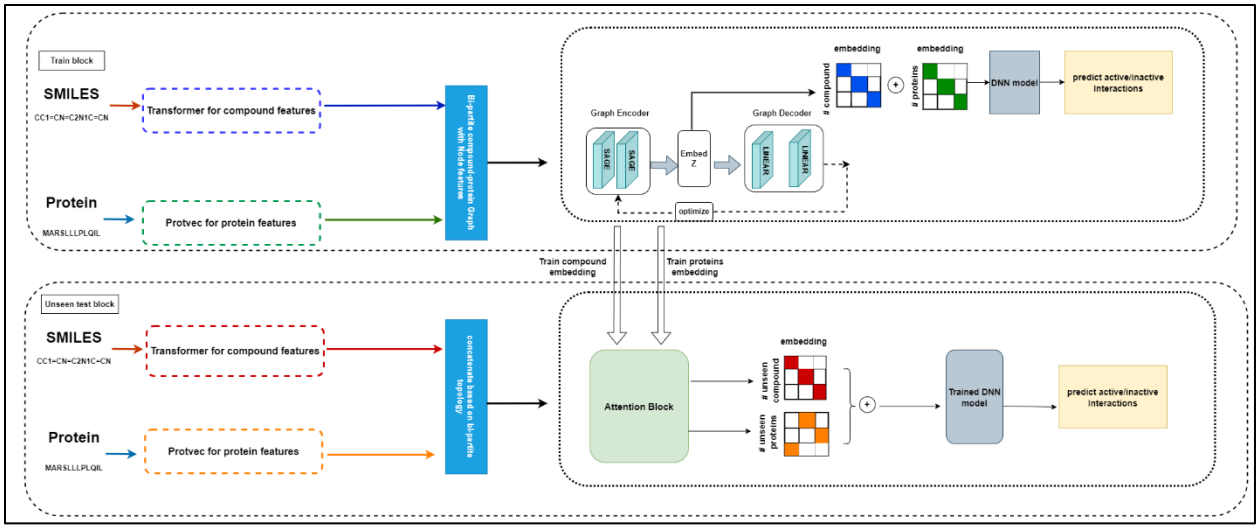

Figure 4-1. The architecture of the GAE2A method. **Train block** includes modules to extract compounds and protein features in an unsupervised manner. The Train block is to train a GAE to provide the node embeddings. **Unseen test block** is designed to utilize the attention mechanism and embeddings of compounds and proteins that participated in the Train block to generate embeddings for unseen compounds and proteins. In the next step, the embeddings concatenate together and use the trained DNN from the Train block to make predictions (for the transductive analysis, just the Train block is applied).

4.2.1.1 Train Block

This block is implemented to handle transductive analysis and is used in the training phase of inductive analysis. It inputs a bi-partite graph with its node features and a GA, including a graph encoder with three graphSAGE layers with the ReLU (90) activation function and a linear layer in the graph decoder to reconstruct the graph. The compound and protein embeddings for each interaction are concatenated with each other as inputs to the DNN layer to do link prediction.

4.2.1.1.1 Compounds and Protein Features

GAA uses two unsupervised methods independently of the training procedure: the pre-trained Transformer (91) and Protvec (92). These models create 512 and 100-dimensional embeddings for

compounds and proteins, respectively. The pre-trained Transformer model uses an encoder to convert any compound string in SMILES format of length M to a matrix of [M * N] where N is the size of the embedding. To use the generated matrix of each compound, we calculate the average of the predicted embeddings. Protvec does not rely on fixed-length feature vectors or handcrafted features. Instead, it works based on word2vec (93) and treats amino acid sequences as sentences, where words are amino acid trigrams, to convert protein sequences into continuous vector representations. These vectors capture the semantic relationships between amino acids based on their co-occurrence patterns in large protein sequence databases.

4.2.1.1.2 GAE

The GA is a DL model that takes advantage of CNNs to learn representations of graph-structured data. The GA can be illustrated by three main sections: the encoder made by sage layers, the latent space that contains the generated embedding of the nodes, and the decoder with linear layers (94) to reconstruct the graph. The encoder section of the GA transforms the input graph data with node features into a lower-dimensional latent space representation. This latent space represents the key features and relationships presented in the graph data. The decoder section, on the other hand, is responsible for reconstructing the original graph data from the learned latent space representation. We optimize the embedding generated with binary cross-entropy loss (BCE loss). The BCE loss function measures the dissimilarity between predicted binary outcomes and actual binary labels as shown below formula:

$$L = -\sum_i \left(y_i \log\left(p_i\right) + (1 - y_i) \log\left(1 - p_i\right)\right), \text{(4-1)}$$

The $y_i$is the true label of the i-th compound–protein pair, $p_i$is its predicted probability by the model.

4.2.1.2 Unseen Test Block

This block is embedded for inductive analysis to perform prediction on unseen test compounds and proteins. The unseen test block receives the initial features of new nodes plus the embedding of compounds and proteins used in the trainset.

4.2.1.2.1 Attention Mechanism

The attention mechanism block computes the embedding for a new compound/protein v by applying the normalized feature similarity function sim(.) as shown below. (The similarity function is Cosine similarity as it is recommended by reference (14).)

$$\mathbf{z}_v = \sum_{u \in \bar{P}} \text{sim}\,(\mathbf{x}_v, \mathbf{x}_u)\mathbf{z}_u, \qquad (4\text{-}2)$$

The $\underline{p}$ is the set of compounds/proteins in the training graph while X_u and Z_u are the features and embeddings of the previously seen compounds/proteins u. Next, the generated embeddings for each compound and protein are concatenated, which creates linked embeddings between the corresponding nodes. These linked embeddings are then used as input for the trained DNN layer to do prediction.

4.2.2 Method 2: GAE2F

The use of KD to do inductive link prediction on heterogeneous graphs was previously introduced as a model named graph2feat (95,96). They proposed a teacher-student approach to do link predictions on homogeneous and heterogeneous graphs. Their focus was on handling link prediction for unseen test nodes that the information about the edges that connect them was not provided. Considering their model, we changed their teacher model from a stack of sage layers to

a full GA that generates optimized embedding for the train set of compounds and proteins. Then optimizes the process of learning the embedding by BCE Loss function. Furthermore, distill the knowledge of graph`s structure properties with a mean squared error (MSE) loss to the student model in offline mode which means that the teacher model is frozen during the training phase. The student model is a simple multilayer perceptron (MLP) that is a stack of linear, dropout, and batch normalization layers and learns the patterns between the link`s embeddings and labels optimized by its loss function. So, in total, the GAE2F model tries to minimize the below loss function during training.

$$L = \sum_{(i,j)\in\varepsilon\cup\varepsilon^-} L_{\text{sup}}\left(\tilde{a}_{i,j}, a_{i,j}\right) + \sum_{v\in V} L_{KD}(\widetilde{\boldsymbol{z_v}}, \boldsymbol{z_v}), \qquad (4\text{-}3)$$

where L_{sup} is the student's link prediction loss (binary cross-entropy), $\tilde{a}_{i,j,}$ is the inner product similarity score between $\tilde{z}_{i,}$, $\tilde{z}_j$ and $a_{i,j} = 1$.

After training for inductive analysis, the model becomes essentially an MLP with no need to deploy the graph topology, and in transductive analysis, only the teacher model is used without needing to deploy the student block. The whole architecture is plotted in **Figure 4-2.**

4.2.2.1 Teacher Model

The teacher model utilized in this context is identical to the GAE model employed in the GAA model. This teacher model generates embeddings for the links within the training set graph, which consists of compounds and proteins.

4.2.2.2 Student Model

The student model receives the simple concatenation of compounds and protein features based on the bi-partite graph topology and generates a matrix of link embeddings by stacking linear and

batch normalization layers shown in **Figure 4-2.** The final embeddings are input to the DNN layer and optimized by BCE loss.

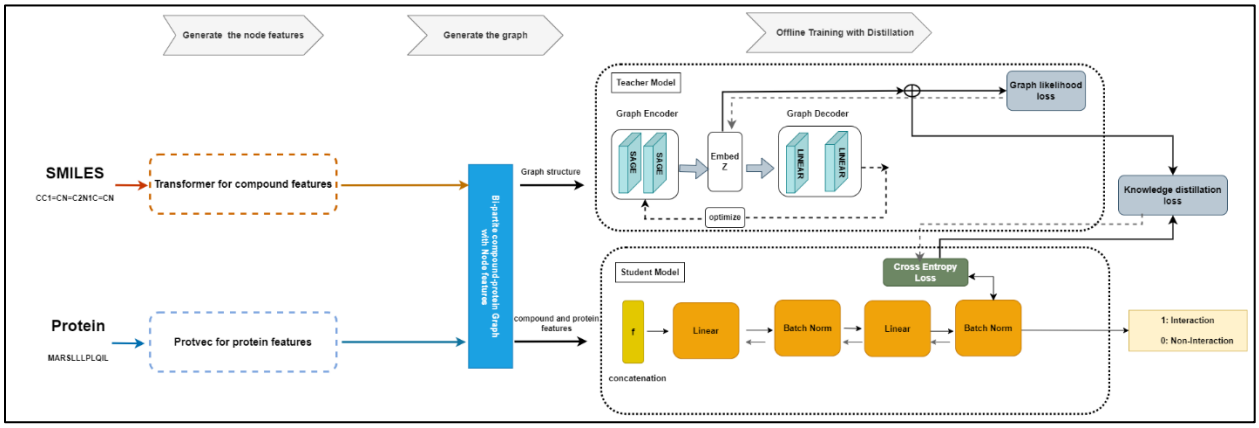

Figure 4-2 The architecture of the GA*E2*F. The first section is designed to generate the node features including modules to extract compounds features and the protein features which works in an unsupervised way. The next section is the generate graph part that is to generate a bi-partite graph. The last part is the offline training with distillation that is to train a classifier with a GA and a MLP to predict the active interaction pairs between compounds and proteins.

4.2.3 Method 3: GraphBAN

In our prior approach, we combined compound and protein features in the student block through simple concatenation. However, recent research (97) has demonstrated that utilizing BAN instead of basic concatenation can significantly enhance CPI prediction accuracy. Additionally, the introduction of a cross-domain adaptation block bridges the gap between in-domain and cross-domain feature distributions, further improving prediction performance. Furthermore, our earlier method involved two separate, unsupervised techniques for extracting compound and protein features. These methods operated independently and had no trainable weights during training.

Previous work indicated that fusing GCN and FCFP features can boost CPI prediction scores (86). We have applied a similar approach to our compound feature generating process. Additionally, for protein feature extraction, we have employed the CNN to convert amino acid sequences into features, following the methodology outlined before (97). Our new approach is named GraphBAN (**Figure 4-3**). It comprises three key components. First, it generates node features by extracting compound and protein features (plus compound feature fusion). Next, it constructs a bi-partite graph with node features. Finally, it employs offline training with distillation using GAE in the student block and BAN layer plus the conditional domain adaptation network (CDAN) module to perform link predictions.

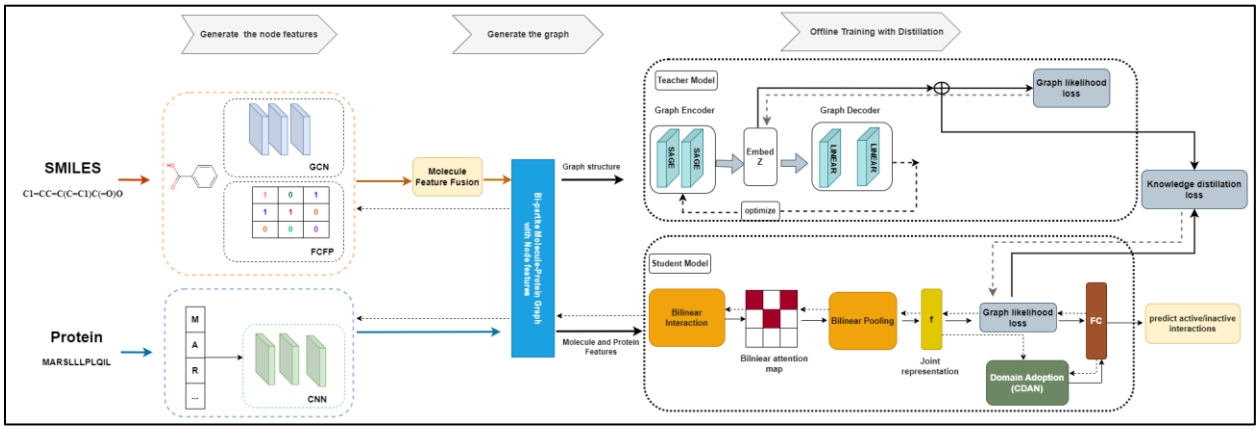

Figure 4-3. The architecture of GraphBAN. The input compound molecule and protein sequences are encoded by (GCNs + FCFP) separately and 1D CNNs. Each row of the encoded compound feature is an aggregated representation of adjacent atoms/bonds plus the substructural features in the compound, and each row of the encoded protein description is a subsequence representation in the protein amino acid sequence.

The compound and protein nodes generate a bi-partite graph. The feature representations are fed into a "Student Model" and the graph structure fed to the "Teacher Model". The "Student Model" includes the Bilinear Interaction plus Bilinear Pooling layers to learn the features pairwise local

interactions. The “Teacher Model” captures the structural features of the graph with a GA module. These two models are connected through a KD loss to share the information from the “Teacher Model” to the “Student Model”. If the prediction mode is on cross-domain, the CDAN module is employed to do a representation alignment between source and domain data.

4.2.3.1 Fusion of GCN and FCFP for Compound Features

In our study, we present a new approach that combines FCFP features with features extracted using a GCN block within a DL model. These combined features are derived from molecular data represented in the SMILES format. To initialize the atom nodes, we utilize the DGL-LifeSci package (98), which assigns a 128-dimensional integer vector to each atom of the compounds, capturing critical chemical properties such as atom type, atom degree, the number of implicit hydrogen atoms, atom hybridization, radical electrons, formal charge, total hydrogen atoms, and aromaticity.

To accommodate compounds of varying sizes, we establish a maximum allowed number of nodes, thereby introducing virtual nodes with zero-padding for smaller compounds. The pivotal component of our approach is the three-layer GCN block, which effectively learns graph representations of compounds. The GCN extends convolutional operators to irregular domains, allowing us to update atom feature vectors by aggregating information from their respective sets of neighboring atoms connected via chemical bonds. This propagation mechanism inherently captures intricate substructure information within the compound, making it a powerful tool for feature extraction.

In a crucial step, we integrate FCFP features with those extracted through GCN processing. This fusion of feature sets enhances the comprehensiveness of our molecular representation, offering a holistic view of the chemical structure. The fusion block works as follows:

$$F_c = F_g + d_{out}\left(\left(t(F_g) * t(F_p)^T\right) * F_p\right), \quad (4\text{-}4)$$

Where $F_c \in R^{n\times 128}$ is the fused compound feature, n is the number of atoms, $F_g \in R^{1\times 128}$ is the molecular feature, $F_p \in R^{1\times 128}$ is the FCFPs feature, and t(.) is the transition function. The initial functional group fingerprint is in the shape of $F_p \in R^{1\times 1024}$ and we use three fully connected layers (with the parameter sizes of 1024 × 512, 512 × 256 and 256 × 128) to reduce the dimensionality of the fingerprint to $F_p \in R^{1\times 128}$. We also add the dropout layer, d_{out}, to reduce network overfitting.

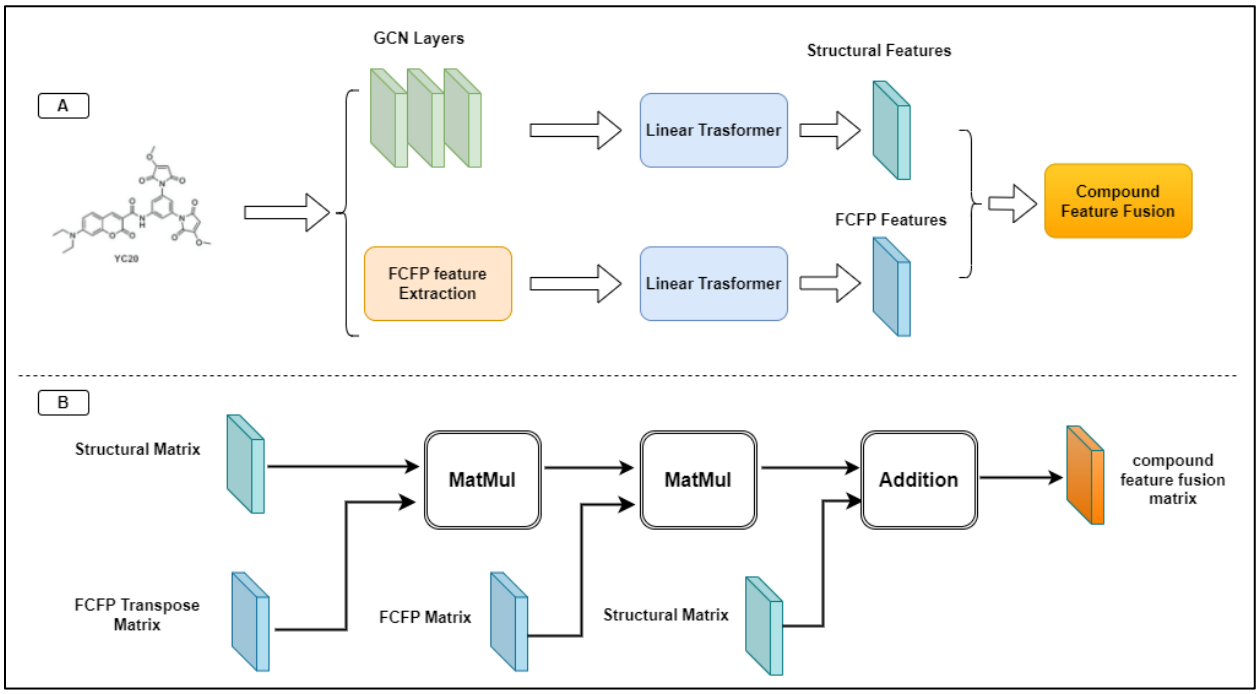

Figure 4-4. Fusion of compound features with FCFP and GCN. **A**. shows how we extract compound features with FCFP and GCN layers and bring them in the same dimensionality with Linear Transform

layers. **B**. Scheme illustrates how we do feature fusion with two "MatMul" layers that operate element-wise multiplication and one "Addition" layer that do the element-wise addition.

4.2.3.2 CNN for Protein Sequence

As shown in **Figure 4-3**, the three-layer CNN block is tailored for protein feature extraction and employs a unique approach that initializes a learnable embedding matrix with representations for all 23 amino acids. This proactive initialization equips the network with essential biochemical information to discern amino acid interactions. Subsequently, the CNN block specializes in extracting local residue patterns from the matrix of protein features, a pivotal step in capturing nuanced dependencies within protein sequences. Notably, this CNN architecture conceptualizes a protein sequence as a sequence of overlapping 3-mer amino acids, enabling it to capture both short-range and long-range interactions within the protein structure. Additionally, the network adheres to a maximum allowed length for protein sequences. The sequences exceeding this length are truncated, while shorter sequences are padded with zeros. This approach reflects a fusion of domain-specific knowledge and deep DL, showcasing its potential for decoding complex protein structures and features while efficiently handling varying sequence lengths. The protein encoder layer is outlined as follows:

$$H_p^{(l+1)} = \sigma\left(CNN\left(W_c^{(l)}, b_c^{(l)}, H_p^{(l)}\right)\right), \quad (4\text{-}5)$$

where $W_c^{(l)}$ and $b_c^{(l)}$ are the learnable weight matrices (filters) and bias vector in the *l*th CNN layer. $H_p^{(l)}$ denotes as $H_p^{(0)} = X_p.\sigma(\cdot)$ where X_p is the corresponding feature matrix for each protein sequence P and the activation function is ReLU(.).

4.2.3.3 Bilinear Attention Neural Network

The BAN which was previously introduced (97) is an integral component of our GraphBAN (showed in **Figures 4-3**). BAN was also previously used in visual question answering problems (99), which proved to be helpful in the CPI task (97). It is designed to capture pairwise local interactions between compounds and proteins. The BAN comprises two key elements: the bilinear interaction map, formed by combining hidden compounds and proteins representations to create an attention-weighted matrix, and the bilinear pooling layer, which extracts a unified compound-protein representation. Pairwise interaction learning is achieved through the bilinear attention mechanism, enhancing the model's predictive capabilities.

4.2.3.3.1 Bilinear Interaction Map

The first component of BAN, the bilinear interaction map, plays a crucial role in modeling the pairwise interactions. It is constructed using the hidden representations of both drugs and targets. This process results in a pairwise interaction matrix, which encapsulates the attention weights assigned to each compound-protein pair. The values in this matrix denote the strength of interaction or relevance between specific compounds and proteins, thereby facilitating the identification of critical compound-protein associations. The single head pairwise interaction matrix $\mathbf{I} \in \mathbb{R}^{M \times N}$ comes from the CNN and compound fusion encoders in the third layer generating the hidden protein and drug representations where M and N show the number of encoded substructures in a amino acid sequence of a protein and atoms/bonds in a compound. The **I** in the ith ang jth column represents as follows:

$$I_{ij} = q^T \left(\sigma\left(U^T h_d^i\right) \circ \sigma\left(V^T h_p^j\right) \right), \quad (4\text{-}6)$$

Where h_d^i is the *i*th column of the last compound fusion block`s layer and h_p^j is the *j*th column of the last layer of the protein CNN block. The $\mathbf{U} \in \mathbb{R}^{D_d \times K}$ and $\mathbf{V} \in \mathbb{R}^{D_p \times K}$ are learnable weight matrices for compound and protein representations, $\mathbf{q} \in \mathbb{R}^K$ is a learnable weight vector, and ∘ denotes element-wise product.

4.2.3.3.2 Bilinear Pooling Layer

The second component of BAN is the pooling layer to obtain the joint representation $\mathbf{f} \in \mathbb{R}^K$, applying over the interaction map **I**. The *k*th element of f' is computed as

$$f'_k = \sigma\left(\left(H_d^{(3)}\right)^\top U\right)_k^\top \cdot I \cdot \sigma\left(\left(H_p^{(3)}\right)^\top V\right)_k$$

$$= \sum_{i=1}^{N} \sum_{j=1}^{M} \mathbf{I}_{i,j}\left(\mathbf{h}_d^i\right)^\top (\mathbf{U}_k \mathbf{V}_k^\top)\mathbf{h}_p^j, \quad (4\text{-}7)$$

where $\mathbf{U}_k$ and $\mathbf{V}_k$ denote the *k*th column of weight matrices **U** and **V**. Moreover, to obtain more compact feature map, we have a sum pooling on the joint representation vector:

$$f = SumPool\,(f', s), \quad (4\text{-}8)$$

where the S*umPool*(·) function is a sum pooling operation with stride s and it converts the dimensionality of $\mathbf{f'} \in \mathbb{R}^K$ to $\mathbf{f} \in \mathbb{R}^{k/s}$**.** In the last step we feed the joint representation **f** into a fully connected layer to classify the inputs and the objective of training is to minimize the Binary Cross-Entropy with Logit loss function.

4.2.3.4 Cross-domain Adaptation to Enhance Generalization

In cross-domain analysis, (**Figure 4-3)**, we tend to train our model with CPIs and to ensure that the trained model can perform well in real-world cases where the compounds/proteins are different from the nodes in the training set based on their features distributions. As a result of this scenario,

it will become hard for simple ML/DL models to perform well on cross-domain data in the test sets. To overcome this distribution shift issue between the train and test data, a model is proposed (100) that uses the CDAN to combine adversarial networks with multilinear feature mapping. As it was demonstrated that using CDAN can improve DTI prediction accuracy (97), we also embed the CDAN into GraphBAN to enhance the performance of inductive cross-domain CPI prediction.

As shown in **Figure 4-5**, the BAN layer generates the source domain joint representation of compound-protein pairs called X_s with true labels Y_s plus the target domain joint as X_t without any label. The CDAN`s workflow starts with the component $f(.)$ as the feature extractor that generates concatenation of separate initial features of the nodes and the BAN`s output for source and target domains separately as follows, $f_s = F(X_s), f_t = F(X_t)$. The next module is a classifier layer called G(.) that works as the generator part in the adversarial loss, that generates $g_s = G(X_s)$ for the source and $g_t = G(X_t)$ for the target data.

To be able to apply the domain discriminator, we need to have the joint conditional representation of the g and f called h:

$$h = FLATTEN\ (f \oplus g), \qquad (4\text{-}9)$$

that $(\oplus)$ is the outer product.

Based on the CDAN`s workflow, we align the joint h for both the source and target domains by the domain discriminator module D(.). The task of D function is to learn how to distinguish between the join representation h generated from the source and target data domains. As the final goal of the conditional adversarial network the $f(.)$ and G(.) functions are trained to minimize the source domain cross entropy loss L by having the true label information and simultaneously

generate h in an indistinguishable way for D function. The following loss functions represent the cross-entropy loss (L_s) and adversarial loss (L_{adv}):

$$L_s(F, G) = E_{(x_i^s, y_i^s)\sim S_s} L\big(G\big(F(x_i^s)\big), y_i^s\big), \quad (4\text{-}10)$$

$$L_{adv}(F, G, D) = E_{x_i^t \sim S_t} log\left(1 - D(f_i^t, g_i^t)\right) + E_{x_j^s \sim S_s} log\left(D\big(f_j^s, g_j^s\big)\right), \quad (4\text{-}11)$$

Following the procedure of optimization for the adversarial loss we need to yield minmax model and define the final representation of CDAN`s loss function (L_{CDAN}) as below:

$$L_{CDAN} = \max_{D} \min_{F,G} L_s(F, G) - \beta L_{adv}(F, G, D), \quad (4\text{-}12)$$

Where $\beta > 0$ is a hyperparameter to weight L_{adv}.

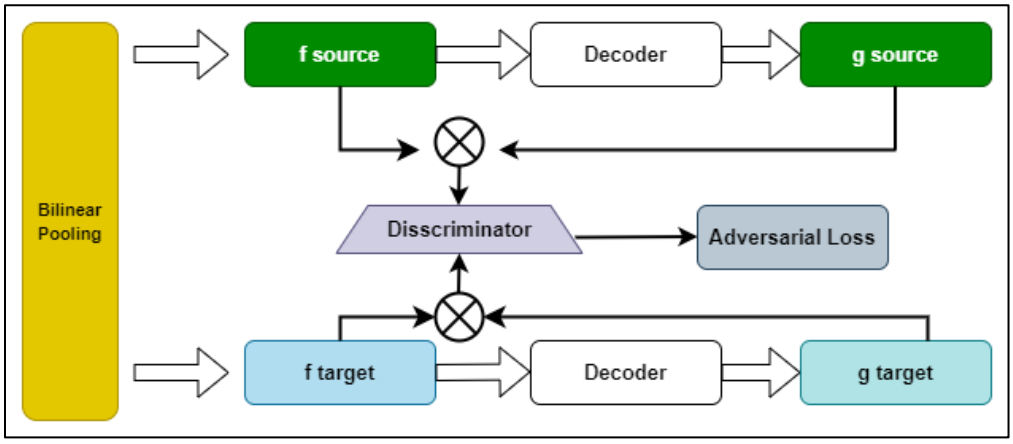

Figure 4-5. The CDAN module. This module is to bring the training and test data distributions to the same. It receives input from the BAN layer, which generates concatenation of compound and protein features and SoftMax logits **g** for source and target domains into a joint conditional representation generated by the discriminator module. The discriminator has two fully connected layers with an adversarial loss to minimize the classification error between the source and target domains.

4.2.3.5 The Loss Functions Implemented in GraphBAN

As depicted in **Figure 4-3**, GraphBAN has four loss functions in total. The first one is the BCE loss that is used in the teacher block, and we use it just for the GA optimization. The second loss is the KD loss, which is the connection between the teacher and student blocks. Based on previous recommendations (95) we use MSE loss and define L_{KD} as follows,

$$L_{KD} = \frac{1}{n}\sum_{i=1}^{n}(T_i - F_i)^2, \quad (4\text{-}13)$$

where the teacher block`s output is $T_i \epsilon R^{i\times e}$ and the joint representation generated by BAN layer is $B_i \epsilon R^{i\times e}$, the $i \in R^{1\times b}$, b is the batch size and e is the embedding size.

The third loss function is the main model`s loss, which is a BCE loss defined as,

$$L_{model} = -\sum_i \left(y_i \log(p_i) + (1-y_i)\log(1-p_i)\right) + \frac{\lambda}{2} \| \boldsymbol{\Theta} \|_2^2 \quad (4\text{-}14)$$

Where θ is the set of all learnable values, y_i is the true label of the ith compound-protein pair, p_i is its predicted output probability by the model and λ is the hyperparameter for L2 regularization.

So far, if we want to use GraphBAN for the transductive analysis we add L_{model} with L_{KD} call it $L_{transductive}$,

$$L_{transductive} = L_{model} + L_{KD}, \quad (4\text{-}15)$$

However, if we want to use the model in inductive mode, we need to add L_{CDAN} to the above equation. So, in inductive analysis the final loss function is defined as follows,

$$L_{inductive} = L_{transductive} + L_{CDAN}, \quad (4\text{-}16)$$

4.2.4 Clustering Strategy

For the cross-domain performance evaluation, we employ a single-linkage clustering method, a bottom-up hierarchical approach. This method ensures that the distances between data in different clusters are always greater than a predefined threshold value, which we denote as γ. This property helps maintain an appropriate separation between clusters, preventing them from being too close to each other.

To represent the data, we utilize two distinct feature extraction methods. For compounds, we use ECFP with a depth of 4 (ECFP4). ECFP4 is a widely used molecular fingerprinting method that captures structural information about compounds. It represents compounds as binary vectors, where each bit in the vector corresponds to the presence or absence of a specific substructure or molecular pattern.

For proteins, we employ protein structure characterization (PSC) features (101). PSC is a feature representation that captures essential characteristics of protein structures, including information about amino acid sequences, secondary structure, and other relevant protein properties. We use cosine distance to measure the similarity between protein structures based on their PSC features. The statistics and results of clustering are added in **Table 4-1**.

Table 4-1. The single-linkage threshold and number of clusters for compounds and proteins based on different datasets.

Dataset	γ threshold	# compound cluster	# protein cluster
BindingDB	2.0	6,084	1,538
BioSNAP	0.8	2,531	2,010
Johnson	1.5	17,773	150

To create source domain data, we randomly select 60% of the compound and protein clusters for each dataset, considering all associated compound-target pairs within these clusters. The remaining 40% of clusters are taken as the target domain data.

4.2.5 Experimental Setting

4.2.5.1 Datasets

We evaluate GraphBan, GAE2A, GAE2F, and four state-of-the-art baselines on three public CPI datasets: BindingDB (102), BioSNAP (103), and Johnson (52). BindingDB is a public database that contains over a million data entries of experimental CPIs with numerical affinities, primarily derived from scientific articles and US patents. As we need a low-biased version of BindingDB, we use a selection of the dataset previously created (36) that includes 14,643 unique compounds and 2,623 proteins. BioSNAP includes 4,510 compounds and 2,181 proteins previously compiled and preprocessed (36). The third dataset (Johnson) is a chemogenomic profile performed in the *Mycobacterium tuberculosis* with links between 47,217 small compounds and around 150 hypomorph or knockdown strains that under express essential proteins. Therefore, the Johnson links mostly represent indirect interactions between compounds and proteins. The Johnson links could include direct CPIs for some of the compounds. After applying pre-processing to this dataset, we have balanced the number of interactions with 22,000 compounds and 149 proteins.

The preprocessing steps for the Johnson dataset are as follows First, we remove the isolated molecules, which are molecules that have no interaction with any of the proteins represented by the *Mycobacterium tuberculosis* hypomorphs. This step results in the removal of approximately

25,000 molecules, reducing the dataset from approximately 7 million to 3 million interactions (22000 × 150 = 3,300,000).

Moreover, as the original dataset with three million interactions is highly unbalanced, with only 1% of the interactions labeled as active or "1" while the rest are labeled as inactive or "0". To address this issue, we applied the under-sampling strategy (104) to remove unnecessary zero interactions, resulting in a balanced dataset with around 815,000 interactions. The statistics of the preprocessed datasets are presented in **Table 4-2**.

Table 4-2. The datasets statistics

Dataset	# Compounds	# Proteins	# Interactions
BindingDB	14,643	2,623	49,199
BioSNAP	4,510	2,181	27,464
Johnson	22,535	150	815,259

4.2.5.2 Implementation

The three proposed methods are developed using Python 3.8 and PyTorch 1.7.1 (105), DGL 0.7.1 (106), DGL-lifeSci 0.2.8, Scikit-learn 1.0.2, Numpy 1.20.2 (107), Pandas 1.2.4 (108), and RDKit 2021.03.2 libraries. The hyperparameters used in the three methods are provided in **Tables 4-3, 4-4, 4-5**. The hyperparameters were selected based on experiment trials and baselines` suggestions.

Table 4-3. GAE2A hyperparameters configuration

Module	Hyperparameter	Value
Transformer for compound features	Embedding size	512

Protvec for protein features	Embedding size	100
Graph encoder for compounds	Embedding size	128
Graph encoder for proteins	Embedding size	128
GAE optimizer	Learning rate	0.001
DNN classifier	Learning rate	0.001
Whole module	Max epochs	200

Table 4-4. GAE2F hyperparameters configuration

Module	Hyperparameter	Value
Transformer for compound features	Embedding size	512
Protvec for protein features	Embedding size	100
Graph encoder for compounds	Embedding size	128
Graph encoder for proteins	Embedding size	128
GAE optimizer	Learning rate	0.001
DNN classifier	Learning rate	0.001
Student	Batch normalization	0.5
Student	Number of linear layers	2
Student	Linea layer sizes	[612, 256]
Whole module	Max epochs	200

Table 4-5. GraphBAN hyperparameters configuration

Module	Hyperparameter	Value
Mini batch	Batch size	64
Three-layer CNN protein encoder	Initial amino acid embedding	128

Three-layer GCN molecule encoder	Initial atom embedding	128
Bilinear interaction attention	Heads of bilinear attention	2
Discriminator	Number of hidden neurons	256
Graph encoder for compounds	Embedding size	128
Graph encoder for proteins	Embedding size	128
GAE optimizer	Learning rate	0.001
Fully connected network	Number of hidden neurons	512
Whole module	Max epochs	50

4.2.5.3 Baselines

Our baseline framework encompasses the RF (109) model, which has input augmented with Transformer-based features, alongside the utilization of Protvec embeddings for the effective representation of compound and protein data. Additionally, we introduce three prominent models: GraphDTA (39), an architecture that leverages GNNs to encode compound molecular graphs and employs CNNs for encoding protein sequences; MolTrans (36), a DL model that innovatively adapts the Transformer network to encode both compound and protein information, enhancing its predictive capabilities through a CNN-based interactive module designed to capture sub-structural interactions. Moreover, we introduce DrugBAN (97), a model rooted in our student framework. The model has no feature fusion module, but it concatenates compound and protein features derived from the GCN and CNN modules, and further incorporates BAN and CDAN for cross-domain analysis without the inclusion of graph analysis within its architecture. These models collectively represent renowned paradigms in the realm of CPI prediction. However, they do not

function in the same way as our proposed methods, as they did not focus on analyzing the input training data as a graph and performing cross-domain analysis.

4.3 Results

4.3.1 Evaluation Strategies and Metrics

We study the models` link prediction (classification) performance on three different datasets, BindingDB, BioSNAP and Johnson. We use two different split strategies: one is transductive, which means the nodes that participate in test set interactions are seen in the training of the model. The strategy splits the links or interactions in the bi-partite graph with compounds and proteins into a 7:2:1 ratio for training, validation and test sets, where the nodes in the validation and test sets are seen in the training data.

The other strategy is cross-domain evaluation which is essential in assessing the robustness and real-world applicability of the models in drug discovery. By testing a model`s performance across different datasets representing diverse compound/protein domains, we can gain a more comprehensive understanding of its generalization capabilities. One such approach for cross-domain evaluation involves a clustering-based pair splitting strategy.

To implement this strategy, we employ a clustering technique introduced previously using the single-linkage algorithm on ECFP4 fingerprints for compounds and PSC for proteins. We then randomly select 60% of the compound clusters and 60% of the protein clusters derived from the clustering step. The compound-protein pairs originating from these selected clusters are designated as the source domain data. Conversely, all pairs occurring between compounds and proteins in the remaining clusters are categorized as target domain data. For our cross-domain evaluation, we

adhere to the standard domain adaptation setting (97), using all labelled source domain data and 80% of the unlabeled target domain data for model training. The remaining 20% of the labelled target domain data form the test set.

We employ a comprehensive approach to assess our model's performance, which includes three key evaluation metrics: the AUROC, the area under the precision-recall curve (AUPRC), and the F1-score. To ensure the reliability of our findings and to account for variability, we conduct five different runs with distinct random seeds and report the average scores obtained from these runs. This averaging of scores provides a more representative and stable measure of our model's performance across various datasets and domains in cross-domain scenarios.

4.3.2 Analysis of Performance on Public Datasets

4.3.2.1 CPI Predictions under Transductive Mode

Here we compare GraphBAN and two other methods we proposed with four other published methods: RF, GraphDTA, MolTrans and DrugBan on the three datasets under transductive mode. In this mode, our train and test set nodes are from the same domain, so we do not need to utilize the CDAN module. In addition, when employing the GAE2A method we do not need to use the "Unseen test block". Finally, when using the GAE2F method, we turn off the "Student Model" as we do not have any inductive links. Of note is that during the transductive analysis, the GAE2A and GAE2F methods are technically the same.

Tables 4-6, 4-7 and 4-8 show the comparative performance of our primary method and the baselines on three different datasets. Our methods outperformed baselines in terms of AUROC, AUPRC, and F1-Score. Moreover, GAE2A/GAE2F outperformed as the best model in BioSNAP

dataset and even around 1.5% higher in AUROC score compared with GraphBAN. Figureure 6 shows the AUROC and AUPRC curves of our proposed methods and other baselines, based on each dataset. The comparison demonstrates that our proposed GraphBAN and GAE2A/GAE2F methods are better than other baselines in terms of transductive prediction performance.

Table 4-6. Transductive analysis on BioSNAP dataset.

Model	AUROC	AUPRC	F1-Score
RF	0.686	0.701	0.690
Graph DTA	0.825	0.840	0.830
MolTrans	0.862	0.865	0.854
DrugBAN	0.901	<u>0.902</u>	<u>0.900</u>
GAE2A/GAE2F	**0.937**	0.857	**0.926**
GraphBAN	<u>0.912</u>	**0.904**	<u>0.900</u>

The best results for each model are marked in bold and the second-best results are underlined.

Table 4-7. Transductive analysis on BindingDB dataset.

Model	AUROC	AUPRC	F1-Score
RF	0.552	0.562	0.550
Graph DTA	0.951	0.932	0.933
MolTrans	0.951	0.933	0.920
DrugBAN	<u>0.960</u>	<u>0.947</u>	<u>0.930</u>

GAE2A/GAE2F	0.933	0.820	0.920
GraphBAN	**0.963**	**0.950**	**0.945**

The best results for each model are marked in bold and the second-best results are underlined.

Table 4-8. Transductive analysis on Johnson dataset

Model	AUROC	AUPRC	F1-Score
RF	0.879	0.880	0.780
Graph DTA	0.870	0.882	0.790
MolTrans	0.880	0.800	0.781
DrugBAN	<u>0.881</u>	<u>0.886</u>	0.802
GAE2A/GAE2F	0.853	0.774	<u>0.822</u>
GraphBAN	**0.921**	**0.922**	**0.843**

The best results for each model are marked in bold and the second-best results are underlined.

Based on the results derived from the GAE2A/GAE2F methods shown in **Tables 4-1, 4-2 and 4-3**, it is evident that the straightforward GAE module can generate high-quality embeddings. These embeddings effectively capture meaningful descriptors for each link and exhibit impressive accuracy in predicting neighbouring links. Furthermore, it becomes apparent that incorporating a KD process and compound feature fusion can enhance the performance of the DrugBAN module within our GraphBAN framework. However, this also suggests that increasing the model's complexity doesn't always translate to improved performance, as demonstrated by the superior results of the GAE2A/GAE2F methods compared to GraphBAN and DrugBAN, particularly in the BioSNAP dataset.

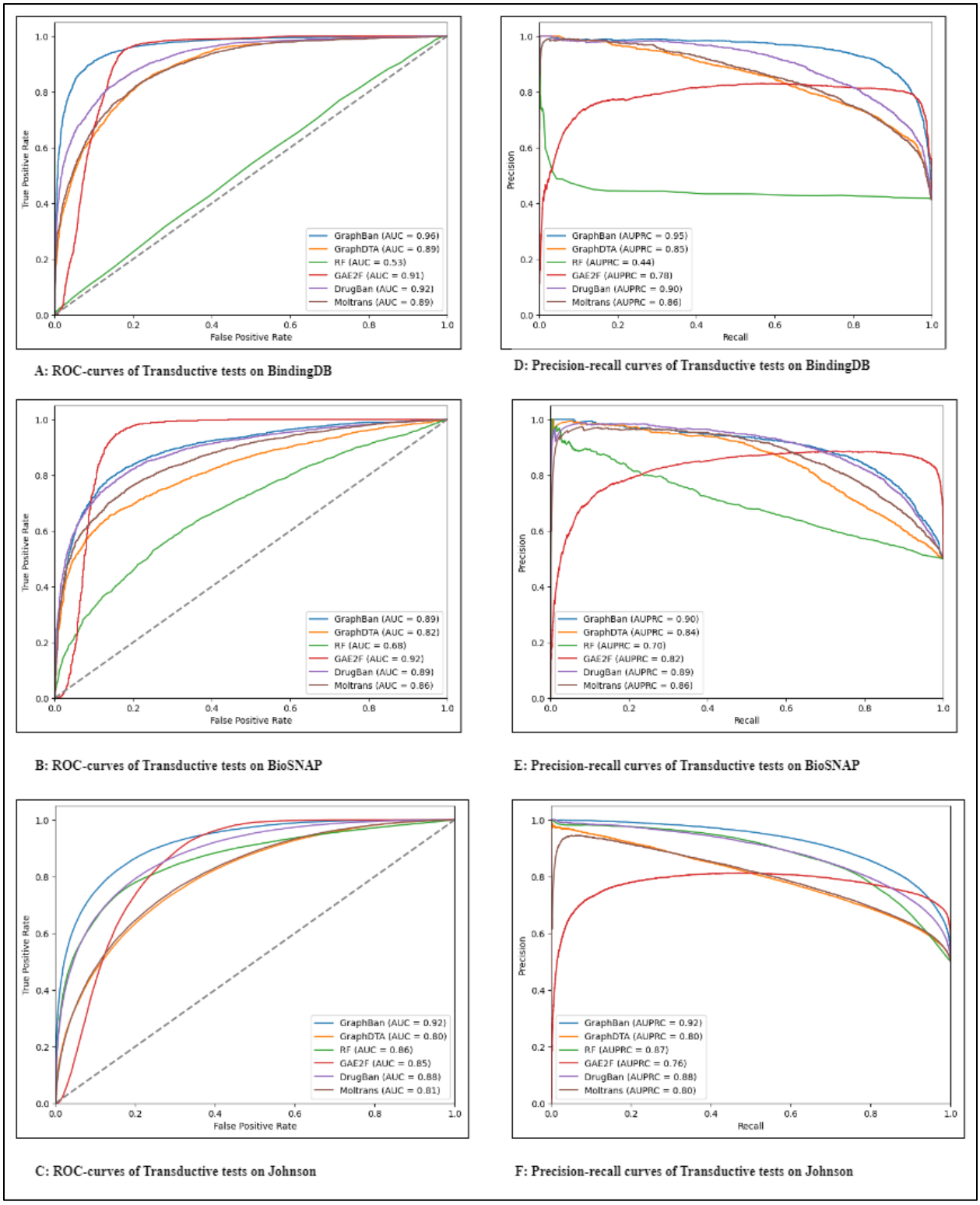

Figure 4-6. AUROC and AUPRC curves under transductive analysis

4.3.2.2 CPI Predictions Under Inductive Mode

In the context of inductivity, as well as the inherent disparities between the distributions of training and test data encountered more frequently in real-world scenarios, we conduct a comparative analysis between our proposed methodologies and the established baselines under these conditions. To address this challenge, we activate the CDAN module within the GraphBAN framework, which enables us to effectively manage the distinct data distributions. Additionally, we employ the previously “Unseen test block” in the GAA model and “Student Module" in the GAE2F model to handle inductive nodes. A comprehensive performance assessment is presented across three distinct datasets, and the results are elaborated upon in **Tables 4-9, 4-10,** and **4-11**, while **Figure 4-7** illustrates the performance comparisons through AUROC and AUPRC curves. The overall results are decreased compared with the transductive analysis, which indicates the challenge of dealing with inductive CPI prediction no matter what models are applied. In line with our findings in transductive analysis, GraphBAN consistently outperforms other state-of-the-art models across the board in inductive analysis. Specifically, it exhibits superior performance compared to DrugBAN, boasting improvements of 0.7%, 2.0%, and 2.2% in AUROC across the BindingDB, BioSNAP, and Johnson datasets. Furthermore, our other proposed methods, GAE2A and GAE2F, outperform the ML-based model (RF), and demonstrate comparable performance with other leading state-of-the-art methods except DrugBAN.

Table 4-9. Inductive analysis on BioSNAP dataset

Model	AUROC	AUPRC	F1-Score
RF	0.622	0.601	0.600
Graph DTA	0.650	0.660	0.655
MolTrans	0.651	0.650	0.652
DrugBAN	<u>0.680</u>	<u>0.731</u>	<u>0.720</u>
GAE2A	0.600	0.611	0.621
GAE2F	0.612	0.615	0.669
GraphBAN	**0.700**	**0.740**	**0.730**

The best results for each model are marked in bold and the second-best results are underlined.

Table 4-10. Inductive analysis on BindingDB dataset

Model	AUROC	AUPRC	F1-Score
RF	0.503	0.495	0.500
Graph DTA	0.660	0.652	0.610
MolTrans	0.684	0.609	0.680
DrugBAN	<u>0.720</u>	<u>0.722</u>	0.708
GAE2A	0.671	0.664	0.651
GAE2F	0.668	0.638	<u>0.718</u>
GraphBAN	**0.727**	**0.737**	**0.720**

The best results for each model are marked in bold and the second-best results are underlined.

Table 4-11. Inductive analysis on Johnson dataset

Model	AUROC	AUPRC	F1-Score
RF	0.601	0.610	0.622
Graph DTA	0.611	0.620	0.628
MolTrans	0.620	0.633	0.630
DrugBAN	<u>0.630</u>	<u>0.651</u>	0.642
GAE2A	0.603	0.623	0.633
GAE2F	0.613	0.632	<u>0.660</u>
GraphBAN	**0.652**	**0.670**	**0.668**

The best results for each model are marked in bold and the second-best results are underlined.

The results provided for inductive CPI prediction indicate that our first model, GAA, is not the top performer. Its performance is comparable to the ML model, RF. However, GAA excels in handling input as a bi-partite graph and fully supports inductive CPI prediction, which is a capability not shared by the other baseline models. On the other hand, the GAE2F method can compete with RF, Graph DTA, and MolTrans as a novel approach in CPI prediction. Nevertheless, it is clear that GAE2F struggles with domain shift between training and test data and KD having only a minimal impact on its performance.

Lastly, looking at our GraphBAN's results, it is evident that incorporating compound feature fusion and graph structural features enhances the model's performance. The differences between GraphBAN and GAE2F lie in adding a BAN layer instead of a simple concatenation, using CDAN

to address different data distributions, and employing supervised-based node features. These modifications are reflected in a considerable improvement in performance, with an average AUROC score across three datasets showing a 6.2% difference. This improvement underscores the considerable influence of the mentioned modules in our model.

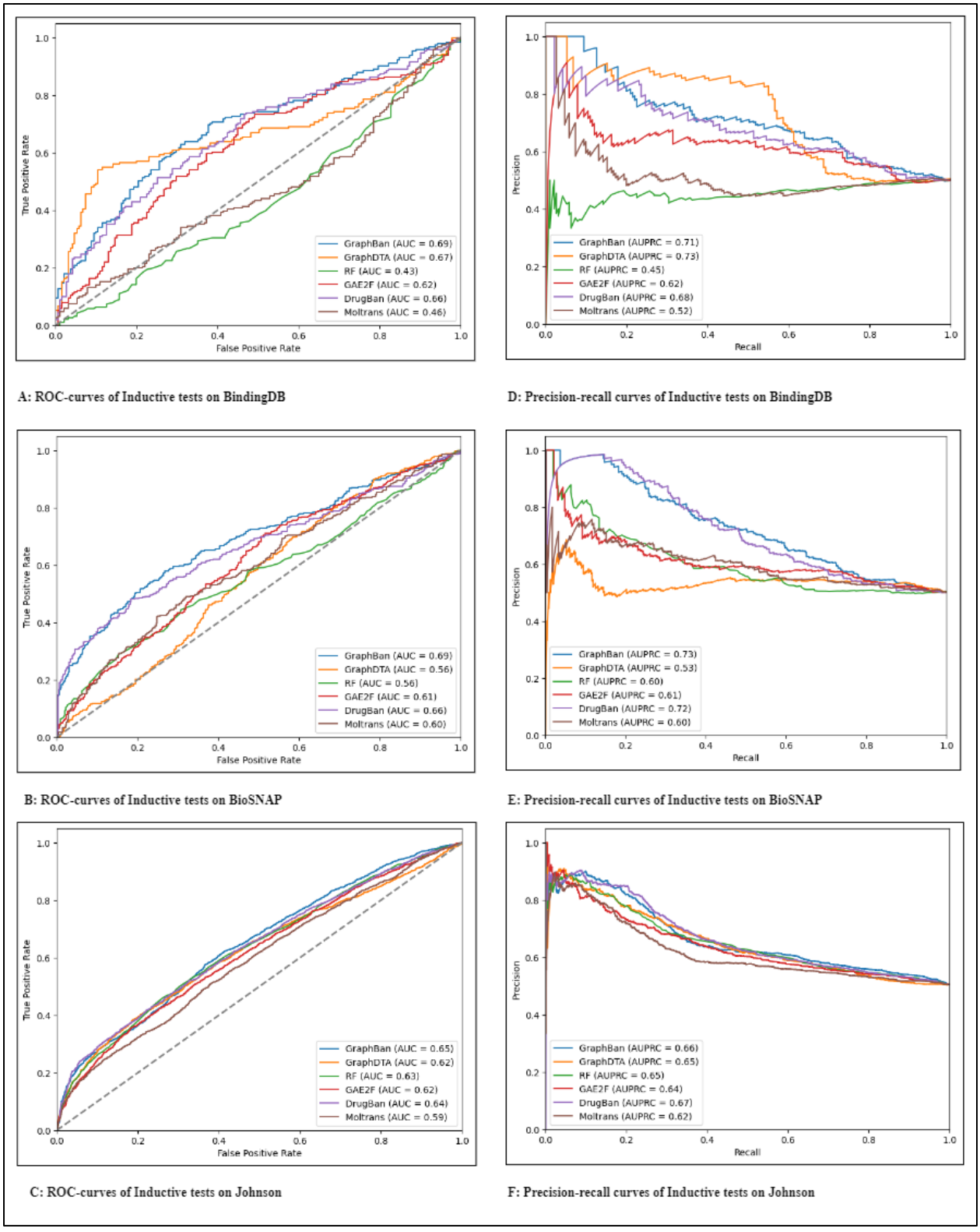

Figure 4-7. AUROC and AUPRC curves under inductive analysis of performance on a case study

4.3.3 Analysis of Performance on a Case Study

To underscore the efficacy and generalization potential of our approach, we predicted 44 ‘unseen” interactions that could be confirmed with previously published literature. This dataset encompasses 44 distinct small compounds and 25 unique proteins. Because the 44 interactions contain only true positive labels, we added 44 interactions with true negative labels for the same protein nodes extracted from the Johnson dataset.

In our experimentation, we find that the BindingDB yields the most favorable prediction results overall. Consequently, we employ the same training set of BindingDB to train the GraphBAN model and subsequently test its performance on the interactions, which closely mirror real-world scenarios and are entirely inductive. The prediction results are aligned with our expectations as we get the following results: AUROC score is 0.65%, AUPRC score is 0.65% and F1-Score is 0.70%. The list of interactions with participating compounds and proteins is provided in **Table 4-12**.

Table 4-12. List of interactions with their true labels that were used in the case study (The references for 44 true positive interactions are provided in the ‘label’ column.)

#	Compound	protein	label	#	compound	protein	Label
1	2471-80	TopA	1 (110)	45	A11617543	TopA	0
2	actinonin	Def	1 (111)	46	A75147472	Def	0
3	amsacrine	TopA	1 (112)	47	A79803969	TopA	0
4	amycobactin	SecY	1 (113)	48	A80061297	SecY	0
5	AN2679	LeuS	1 (114)	49	A90396468	LeuS	0
6	azaserine	GltB	1 (115)	50	K01047527	GltB	0
7	BB-3497	Def	1 (111)	51	K01577834	Def	0
8	Butein	Fas	1 (116)	52	K08797482	Fas	0

9	CCA34	FadD32	1 (117)	53	K08959788	FadD32	0
10	Compound 1_gurcha	AspS	1 (118)	54	K08980087	AspS	0
11	Compound 1_buchieri	CanB	1 (119)	55	K11695342	CanB	0
12	Compound 1_murugesan	NdhA	1 (120)	56	K12602264	NdhA	0
13	Compound 14_jeankumar	GyrB	1 (121)	57	K16297821	GyrB	0
14	Compound 14_krishna	IlvC	1 (122)	58	K20944421	IlvC	0
15	Compound 14_palencia	LeuS	1 (114)	59	K21948397	LeuS	0
16	Compound 15	TrmD	1 (123)	60	K24877262	TrmD	0
17	Compound 16	IlvC	1 (122)	61	K27435692	IlvC	0
18	Compound 18b	Dxr	1 (124)	62	K29653246	Dxr	0
19	Compound 29e	TrmD	1 (125)	63	K30978330	TrmD	0
20	Compound 4c	Def	1 (126)	64	K37960868	Def	0
21	Compound 7	Fum	1 (127)	65	K38851002	Fum	0
22	D155931	Dlat	1 (128)	66	K39792319	Dlat	0
23	DC-159a	GyrA	1 (129)	67	K41217525	GyrA	0
24	Fc14-584B	CanB	1 (130)	68	K46807985	CanB	0
25	GSK85A	AspS	1 (131)	69	K48257186	AspS	0
26	GSK92A	AspS	1 (131)	70	K51916742	AspS	0
27	GSK93A	AspS	1 (131)	71	K54790791	AspS	0
28	GSK97C	AspS	1 (131)	72	K57755201	AspS	0
29	IMT007	GlgB	1 (132)	73	K59574735	GlgB	0
30	LBK-611	Def	1 (111)	74	K62025857	Def	0
31	MB16695	GlgB	1 (132)	75	K62228400	GlgB	0
32	Octoclothepin	ParA	1 (133)	76	K65183503	ParA	0
33	SC-5217501	GuaB2	1 (134)	77	K65659695	GuaB2	0
34	SC-6655281	GuaB2	1 (134)	78	K80789558	GuaB2	0
35	SC-7759844	GuaB2	1 (134)	79	K83932612	GuaB2	0
36	thiolactomycin	KasA	1 (135)	80	K84994806	KasA	0
37	THT-1	DfrA	1 (136)	81	K86290588	DfrA	0

38	VCC234718	GuaB2	1 (137)	82	K87885489	GuaB2	0
39	VXc-486	GyrB	1 (138)	83	K88549378	GyrB	0
40	Z0933	ProB	1 (139)	84	K89202613	ProB	0
41	IDR-0106878	Eno	1 (140)	85	K91045118	Eno	0
42	secneolitsine	TopA	1 (141)	86	K94190539	TopA	0
43	TPSA	GlmU	1 (142)	87	K96007995	GlmU	0
44	GSK3011724A	KasA	1 (143)	88	K99032519	KasA	0

The list of true positive interactions gathered by reference (3)

4.4 Discussion

In this work, we successfully tackled the CPI prediction problem with a KD-based approach, namely GraphBAN. In GraphBAN, we designed embedded modules to participate in a training (backpropagation) process to extract compound and protein features to generate a bi-partite graph with node features, use GAE to extract structural features of the graph and BAN to concatenate nodes` features. These two modules are connected in the form of the KD technique. Experimental results showed that our proposed combination of compound feature fusion, GAE, BAN, and CDAN modules can improve predictions by extracting more informative features, more meaningful concatenation, and taking care of distribution shifts between the training and test data. We conducted comprehensive studies on BioSNAP, BindingDB, and Johnson datasets, showing that our proposed GraphBAN outperformed other baselines on almost all of the evaluation metrics, considering differences between our datasets, such as their size from small to large and their diverse sources. With respect to our inductive analysis, the overall predictive results for all methods are not particularly high. Therefore, more work will be necessary to advance research in this field.

Another area for improvement lies in the two approaches to performing inductive CPI prediction: the semi-inductive approach, where one of the compounds or proteins is absent, and fully inductive, where both nodes are absent. The latter is considerably more challenging for the model to predict the CPIs. Hence, we conducted a fully inductive analysis to demonstrate our model's generalization capabilities. However, as our case study analysis reveals, when both nodes for a CPI interaction are absent and entirely unseen by our model, determining the optimal cutoff for predicted values is impossible. This is compounded by the absence of true labels in real-world scenarios, making it highly likely that we will not select the best cutoff. In contrast, the semi-inductive approach offers the advantage of having one fixed node (e.g., protein), allowing us to make cutoff decisions based on the protein types involved in both the training and testing procedures.

4.5 Summary

In conclusion, our contribution to the field of CPI prediction is encapsulated in the end-to-end DL model, GraphBAN. This model adeptly processes CPI data in both bi-partite graph and tabular formats, enabling transductive and inductive link prediction across cross-domain and in-domain scenarios. GraphBAN incorporates KD, where a GAE serves as a teacher module, imparting structural and neighboring information from the CPI graph to a student model. The student model effectively concatenates the separated features of compounds and proteins involved in interactions using the BAN layer. Furthermore, GraphBAN addresses the distribution gap between training data and inductive cross-domain test data through the CDAN module.

Our proposed GraphBAN exhibits outstanding performance by providing acceptable CPI prediction accuracy, surpassing two other methods we introduced, and four other state-of-the-art methods found in the literature when they are evaluated across three distinct datasets. This research offers a significant advancement in CPI prediction, promising substantial practical implications in drug discovery.

Chapter 5 Conclusion and Future Work

5.1 Conclusion

This thesis presented two significant contributions to the fields of computational chemistry and drug discovery. The first, detailed in Chapter 3, involved the development of a novel DL-based model for clustering small molecules at a large scale. This model successfully integrated both local and global molecular features through PCA and VAE, enabling the effective clustering of molecules from a large chemical library. The second contribution, explored in Chapter 4, was the development of GraphBAN, a graph-based DL model for accurately predicting CPIs in both inductive and transductive settings.

The first study demonstrated how ML techniques could optimize the process of clustering large molecule datasets using advanced embedding strategies that has potential applications in narrowing down the candidates for drug screening, thus saving time and resources.

The second study introduced a novel approach to CPI prediction using graph KD, which is critical for identifying potential drug compounds. GraphBAN, with its unique architecture, proved effective in handling complex data in both bipartite graph and tabular formats, and outperformed several baseline models in various settings.

5.2 Future Work

For future research, several directions can be taken to build on this thesis:

1. Advanced Molecular Clustering Techniques: Explore deep learning models that include dynamic aspects of molecular structures, like molecular dynamics simulations, to better understand molecular behavior over time and potentially find new drug candidates.

2. Expansion of GraphBAN Capabilities: Extend GraphBAN to use multiple types of data, such as genomic, proteomic, and metabolomic, to improve its predictive accuracy and offer a more complete view of CPIs.

3. GraphBAN in Personalized Medicine: Adapt GraphBAN for personalized medicine, focusing on predicting patient-specific drug responses based on genetic variations.

4. Scalable Computational Frameworks: Develop more scalable and efficient computational frameworks for handling large and complex datasets, possibly using parallel computing or cloud-based platforms to improve data processing.

References

1. Patel L, Shukla T, Huang X, Ussery DW, Wang S. Machine Learning Methods in Drug Discovery. Molecules. 2020 Nov 12;25(22):5277.

2. Dara S, Dhamercherla S, Jadav SS, Babu CM, Ahsan MJ. Machine Learning in Drug Discovery: A Review. Artif Intell Rev. 2022;55(3):1947–99.

3. Nag S, Baidya ATK, Mandal A, Mathew AT, Das B, Devi B, et al. Deep learning tools for advancing drug discovery and development. 3 Biotech. 2022 May;12(5):110.

4. Vamathevan J, Clark D, Czodrowski P, Dunham I, Ferran E, Lee G, et al. Applications of machine learning in drug discovery and development. Nat Rev Drug Discov. 2019 Jun 11;18(6):463–77.

5. Glavatskikh M, Leguy J, Hunault G, Cauchy T, Da Mota B. Dataset's chemical diversity limits the generalizability of machine learning predictions. J Cheminform. 2019 Dec 12;11(1):69.

6. Wang X, Liu M, Zhang L, Wang Y, Li Y, Lu T. Optimizing Pharmacokinetic Property Prediction Based on Integrated Datasets and a Deep Learning Approach. J Chem Inf Model. 2020 Oct 26;60(10):4603–13.

7. Wu Z, Ramsundar B, Feinberg EN, Gomes J, Geniesse C, Pappu AS, et al. MoleculeNet: a benchmark for molecular machine learning. Chem Sci. 2018;9(2):513–30.

8. Wan F, Zeng J. Deep learning with feature embedding for compound-protein interaction prediction. Available from: https://doi.org/10.1101/086033

9. Frantzi M, Latosinska A, Mischak H. Proteomics in Drug Development: The Dawn of a New Era? Proteomics Clin Appl. 2019 Mar 25;13(2).

10. Batool M, Ahmad B, Choi S. A Structure-Based Drug Discovery Paradigm. Int J Mol Sci. 2019 Jun 6;20(11):2783.

11. Cobanoglu MC, Liu C, Hu F, Oltvai ZN, Bahar I. Predicting Drug–Target Interactions Using Probabilistic Matrix Factorization. J Chem Inf Model. 2013 Dec 23;53(12):3399–409.

12. Shim JS, Liu JO. Recent Advances in Drug Repositioning for the Discovery of New Anticancer Drugs. Int J Biol Sci. 2014;10(7):654–63.

13. Weininger D. SMILES, a chemical language and information system. 1. Introduction to methodology and encoding rules. J Chem Inf Comput Sci. 1988 Feb 1;28(1):31–6.

14. Rogers D, Hahn M. Extended-Connectivity Fingerprints. J Chem Inf Model. 2010 May 24;50(5):742–54.

15. Landrum, G. et al. RDKit: open-source cheminformatics. https:// github.com/rdkit/rdkit (2006).

16. Mauri A, Consonni V, Pavan M, Todeschini R, others. Dragon software: An easy approach to molecular descriptor calculations. Match. 2006;56(2):237–48.

17. Consonni V, Todeschini R. Molecular Descriptors. In 2010. p. 29–102.

18. Duvenaud DK, Maclaurin D, Iparraguirre J, Bombarell R, Hirzel T, Aspuru-Guzik A, et al. Convolutional Networks on Graphs for Learning Molecular Fingerprints. In: Cortes C, Lawrence N, Lee D, Sugiyama M, Garnett R, editors. Advances in Neural Information Processing Systems [Internet]. Curran Associates, Inc.; 2015. Available from: https://proceedings.neurips.cc/paper_files/paper/2015/file/f9be311e65d81a9ad8150a60844bb94c-Paper.pdf

19. Goh GB, Hodas NO, Siegel C, Vishnu A. SMILES2Vec: An Interpretable General-Purpose Deep Neural Network for Predicting Chemical Properties. 2017 Dec 5;

20. Hawkins PCD, Skillman AG, Warren GL, Ellingson BA, Stahl MT. Conformer Generation with OMEGA: Algorithm and Validation Using High Quality Structures from the Protein Databank and Cambridge Structural Database. J Chem Inf Model. 2010 Apr 26;50(4):572–84.

21. Gharakhanian E, Takahashi J, Clever J, Kasamatsu H. In vitro assay for protein-protein interaction: carboxyl-terminal 40 residues of simian virus 40 structural protein VP3 contain a determinant for interaction with VP1. Proceedings of the National Academy of Sciences. 1988 Sep;85(18):6607–11.

22. Ebrahimi M, Lakizadeh A, Agha-Golzadeh P, Ebrahimie E, Ebrahimi M. Prediction of Thermostability from Amino Acid Attributes by Combination of Clustering with Attribute Weighting: A New Vista in Engineering Enzymes. PLoS One. 2011 Aug 10;6(8):e23146.

23. Wang J, Cao H, Zhang JZH, Qi Y. Computational Protein Design with Deep Learning Neural Networks. Sci Rep. 2018 Apr 20;8(1):6349.

24. Mu Z, Yu T, Liu X, Zheng H, Wei L, Liu J. FEGS: a novel feature extraction model for protein sequences and its applications. BMC Bioinformatics. 2021 Jun 3;22(1):297.

25. Zhang J, Liu B. A Review on the Recent Developments of Sequence-based Protein Feature Extraction Methods. Curr Bioinform. 2019 Mar 7;14(3):190–9.

26. Liu S, Cui C, Chen H, Liu T. Ensemble learning-based feature selection for phosphorylation site detection. Front Genet. 2022 Oct 21;13.

27. Zhang J, Liu B. A Review on the Recent Developments of Sequence-based Protein Feature Extraction Methods. Curr Bioinform. 2019 Mar 7;14(3):190–9.

28. Zhang YF, Wang X, Kaushik AC, Chu Y, Shan X, Zhao MZ, et al. SPVec: A Word2vec-Inspired Feature Representation Method for Drug-Target Interaction Prediction. Front Chem. 2020 Jan 10;7.

29. Grechishnikova D. Transformer neural network for protein-specific de novo drug generation as a machine translation problem. Sci Rep. 2021 Jan 11;11(1):321.

30. Yang F, Liu J, Zhang Q, Yang Z, Zhang X. CNN-based two-branch multi-scale feature extraction network for retrosynthesis prediction. BMC Bioinformatics. 2022 Sep 2;23(1):362.

31. Wu S, Sun F, Zhang W, Xie X, Cui B. Graph Neural Networks in Recommender Systems: A Survey. 2020 Nov 4;

32. Zhang M, Chen Y. Link prediction based on graph neural networks. Adv Neural Inf Process Syst. 2018;31.

33. Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, et al. Attention is All you Need. In: Guyon I, Luxburg U Von, Bengio S, Wallach H, Fergus R, Vishwanathan S, et al., editors. Advances in Neural Information Processing Systems [Internet]. Curran Associates, Inc.; 2017. Available from: https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf

34. Kim H, Lee J, Ahn S, Lee JR. A merged molecular representation learning for molecular properties prediction with a web-based service. Sci Rep. 2021 May 26;11(1):11028.

35. Grechishnikova D. Transformer neural network for protein-specific de novo drug generation as a machine translation problem. Sci Rep. 2021 Jan 11;11(1):321.

36. Huang K, Xiao C, Glass LM, Sun J. MolTrans: molecular interaction transformer for drug–target interaction prediction. Bioinformatics. 2021 Mar 15;37(6):830-6.

37. Maziarka Ł, Danel T, Mucha S, Rataj K, Tabor J, Jastrzębski S. Molecule Attention Transformer. 2020 Feb 19;

38. Jumper J, Evans R, Pritzel A, Green T, Figurnov M, Ronneberger O, et al. Highly accurate protein structure prediction with AlphaFold. Nature. 2021 Aug 26;596(7873):583–9.

39. Nguyen T, Le H, Quinn TP, Nguyen T, Le TD, Venkatesh S. GraphDTA: predicting drug-target binding affinity with graph neural networks. Available from: https://doi.org/10.5281/zenodo.3603523.

40. Macarron R, Banks MN, Bojanic D, Burns DJ, Cirovic DA, Garyantes T, et al. Impact of high-throughput screening in biomedical research. Nat Rev Drug Discov. 2011 Nov;10(3):188–95.

41. Zhong F, Xing J, Li X, Liu X, Fu Z, Xiong Z, et al. Artificial intelligence in drug design. Sci China Life Sci. 2018 Nov;61(10):1191–204.

42. Hochreiter S, Klambauer G, Rarey M. Machine Learning in Drug Discovery. J Chem Inf Model. 2018 Nov;58(9):1723–4.

43. Wu Z, Ramsundar B, Feinberg EN, Gomes J, Geniesse C, Pappu AS, et al. MoleculeNet: a benchmark for molecular machine learning. Chem Sci. 2018;9(2):513–30.

44. Yang K, Swanson K, Jin W, Coley C, Eiden P, Gao H, et al. Analyzing Learned Molecular Representations for Property Prediction. J Chem Inf Model. 2019 Nov;59(8):3370–88.

45. Sivaraman G, Jackson NE, Sanchez-Lengeling B, Vázquez-Mayagoitia Á, Aspuru-Guzik A, Vishwanath V, et al. A machine learning workflow for molecular analysis: application to melting points. Mach Learn Sci Technol. 2020 Nov;1(2):25015.

46. Jiang D, Wu Z, Hsieh CY, Chen G, Liao B, Wang Z, et al. Could graph neural networks learn better molecular representation for drug discovery? A comparison study of descriptor-based and graph-based models. J Cheminform. 2021 Nov;13(1):12.

47. Rogers D, Hahn M. Extended-connectivity fingerprints. J Chem Inf Model. 2010;50(5):742–54.

48. David L, Thakkar A, Mercado R, Engkvist O. Molecular representations in AI-driven drug discovery: a review and practical guide. J Cheminform. 2020 Nov;12(1):56.

49. Camacho DM, Collins KM, Powers RK, Costello JC, Collins JJ. Next-Generation Machine Learning for Biological Networks. Cell. 2018 Nov;173(7):1581–92.

50. Karim MR, Beyan O, Zappa A, Costa IG, Rebholz-Schuhmann D, Cochez M, et al. Deep learning-based clustering approaches for bioinformatics. Brief Bioinform. 2021 Nov;22(1):393–415.

51. Alizadeh AA, Eisen MB, Davis RE, Ma C, Lossos IS, Rosenwald A, et al. Distinct types of diffuse large B-cell lymphoma identified by gene expression profiling. Nature. 2000 Nov;403(6769):503–11.

52. Johnson EO, LaVerriere E, Office E, Stanley M, Meyer E, Kawate T, et al. Large-scale chemical–genetics yields new M. tuberculosis inhibitor classes. Nature. 2019 Nov;571(7763):72–8.

53. Landrum G. RDKit: Open-source cheminformatics [Internet]. Available from: http://www.rdkit.org

54. Ringnér M. What is principal component analysis? Nat Biotechnol. 2008 Nov;26(3):303–4.

55. Guha R, Willighagen E. A Survey of Quantitative Descriptions of Molecular Structure. Curr Top Med Chem. 2012;12(18):1946–56.

56. Seger C. An investigation of categorical variable encoding techniques in machine learning: binary versus one-hot and feature hashing [Internet]. 2018. Available from: http://urn.kb.se/resolve?urn=urn:nbn:se:kth:diva-237426

57. Gilmer J, Schoenholz SS, Riley PF, Vinyals O, Dahl GE. Neural Message Passing for Quantum Chemistry. arXiv:170401212 [cs] [Internet]. 2017 Nov; Available from: http://arxiv.org/abs/1704.01212

58. Zhou J, Cui G, Zhang Z, Yang C, Liu Z, Wang L, et al. Graph Neural Networks: A Review of Methods and Applications. arXiv:181208434 [cs, stat] [Internet]. 2019 Nov; Available from: http://arxiv.org/abs/1812.08434

59. Jolliffe IT, Cadima J. Principal component analysis: a review and recent developments. Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences. 2016 Nov;374(2065):20150202.

60. MacQueen J. Some methods for classification and analysis of multivariate observations. Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Statistics. 1967 Nov;281–97.

61. Zhang T, Ramakrishnan R, Livny M. BIRCH: A New Data Clustering Algorithm and Its Applications. Data Min Knowl Discov. 1997 Nov;1(2):141–82.

62. Kramer MA. Nonlinear principal component analysis using autoassociative neural networks. AIChE Journal. 1991;37(2):233–43.

63. Geddes TA, Kim T, Nan L, Burchfield JG, Yang JYH, Tao D, et al. Autoencoder-based cluster ensembles for single-cell RNA-seq data analysis. BMC Bioinformatics. 2019 Nov;20(19):660.

64. Kingma DP, Welling M. Auto-Encoding Variational Bayes. arXiv:13126114 [cs, stat] [Internet]. 2014 Nov; Available from: http://arxiv.org/abs/1312.6114

65. Rousseeuw PJ. Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. J Comput Appl Math. 1987 Nov;20:53–65.

66. Caliński T, Harabasz J. A dendrite method for cluster analysis. Communications in Statistics. 1974 Nov;3(1):1–27.

67. Davies DL, Bouldin DW. A Cluster Separation Measure. IEEE Trans Pattern Anal Mach Intell. 1979 Nov;PAMI-1(2):224–7.

68. Pedregosa F, Varoquaux G, Gramfort A, Michel V, Thirion B, Grisel O, et al. Scikit-learn: Machine learning in Python. the Journal of machine Learning research. 2011;12:2825–30.

69. Hinton G, Roweis ST. Stochastic neighbor embedding. In: NIPS. Citeseer; 2002. p. 833–40.

70. Van Der Maaten L, Hinton G. Visualizing Data using t-SNE. Vol. 9, Journal of Machine Learning Research. 2008.

71. Dice LR. Measures of the Amount of Ecologic Association Between Species. Ecology. 1945;26(3):297–302.

72. Rogers DJ, Tanimoto TT. A Computer Program for Classifying Plants. Science (1979). 1960;132(3434):1115–8.

73. Riniker S, Landrum GA. Similarity maps - a visualization strategy for molecular fingerprints and machine-learning methods. J Cheminform. 2013 Nov;5(1):43.

74. Bajusz D, Rácz A, Héberger K. Why is Tanimoto index an appropriate choice for fingerprint-based similarity calculations? J Cheminform. 2015 Nov;7(1):20.

75. Chen L, Wang W, Zhai Y, Deng M. Deep soft K-means clustering with self-training for single-cell RNA sequence data. NAR Genom Bioinform [Internet]. 2020 Nov;2(2). Available from: https://doi.org/10.1093/nargab/lqaa039

76. Tian K, Shao M, Wang Y, Guan J, Zhou S. Boosting compound-protein interaction prediction by deep learning. Methods. 2016 Nov 1;110:64-72.

77. Kuhn P, Wilson K, Patch MG, Stevens RC. The genesis of high-throughput structure-based drug discovery using protein crystallography. Current opinion in chemical biology. 2002 Oct 1;6(5):704-10.

78. Bredel M, Jacoby E. Chemogenomics: an emerging strategy for rapid target and drug discovery. Nat Rev Genet. 2004 Apr;5(4):262–75.

79. Liu C, Hogan AM, Sturm H, Khan MW, Islam MdM, Rahman ASMZ, et al. Deep learning-driven prediction of drug mechanism of action from large-scale chemical-genetic interaction profiles. J Cheminform. 2022 Dec 12;14(1):12.

80. Sieg J, Flachsenberg F, Rarey M. In need of bias control: evaluating chemical data for machine learning in structure-based virtual screening. Journal of chemical information and modeling. 2019 Mar 5;59(3):947-61.

81. Wang W, Yang X, Wu C, Yang C. CGINet: graph convolutional network-based model for identifying chemical-gene interaction in an integrated multi-relational graph. BMC Bioinformatics. 2020 Dec 1;21(1).

82. Yu L, Qiu W, Lin W, Cheng X, Xiao X, Dai J. HGDTI: predicting drug–target interaction by using information aggregation based on heterogeneous graph neural network. BMC Bioinformatics. 2022 Dec 1;23(1).

83. Bian J, Zhang X, Zhang X, Xu D, Wang G. MCANet: shared-weight-based MultiheadCrossAttention network for drug–target interaction prediction. Brief Bioinform. 2023 Mar 1;24(2).

84. Hua Y, Song X, Feng Z, Wu X. MFR-DTA: a multi-functional and robust model for predicting drug–target binding affinity and region. Bioinformatics. 2023 Feb 1;39(2).

85. Hua Y, Song X, Feng Z, Wu XJ, Kittler J, Yu DJ. CPInformer for Efficient and Robust Compound-Protein Interaction Prediction. IEEE/ACM Trans Comput Biol Bioinform. 2023 Jan 1;20(1):285–96.

86. Lee I, Keum J, Nam H. DeepConv-DTI: Prediction of drug-target interactions via deep learning with convolution on protein sequences. PLoS Comput Biol. 2019 Jun 14;15(6):e1007129.

87. Abbasi K, Razzaghi P, Poso A, Amanlou M, Ghasemi JB, Masoudi-Nejad A. DeepCDA: Deep cross-domain compound-protein affinity prediction through LSTM and convolutional neural networks. Bioinformatics. 2020 Sep 1;36(17):4633–42.

88. Kao PY, Kao SM, Huang NL, Lin YC. Toward drug-target interaction prediction via ensemble modeling and transfer learning. In2021 IEEE International Conference on Bioinformatics and Biomedicine (BIBM) 2021 Dec 9 (pp. 2384-2391). IEEE.

89. Arora V, Sanguinetti G. De novo prediction of RNA-protein interactions with Graph Neural Networks. RNA. 2022 Aug 25;rna.079365.122.

90. Agarap AF. Deep Learning using Rectified Linear Units (ReLU). 2018 Mar 22; Available from: http://arxiv.org/abs/1803.08375

91. Morris P, St Clair R, Barenholtz E, Edward Hahn W. Predicting Binding from Screening Assays with Transformer Network Embeddings.

92. Asgari E, Mofrad MRK. Continuous Distributed Representation of Biological Sequences for Deep Proteomics and Genomics. PLoS One [Internet]. 2015;10(11):141287. Available from: http://llp.berkeley.eduandHarvardDataverse:http://dx.doi.org/10.7910/DVN/JMFHTN.

93. Mikolov T, Chen K, Corrado G, Dean J. Efficient Estimation of Word Representations in Vector Space. 2013 Jan 16; Available from: http://arxiv.org/abs/1301.3781

94. Kipf TN, Welling M. Variational Graph Auto-Encoders 1 A latent variable model for graph-structured data. 2016.

95. Samy AE, Kefato ZT, Girdzijauskas S. Graph2Feat: Inductive Link Prediction via Knowledge Distillation. In: ACM Web Conference 2023 - Companion of the World Wide Web Conference, WWW 2023. Association for Computing Machinery, Inc; 2023. p. 805–12.

96. Zhang S, Liu Y, Sun Y, Shah N. Graph-less neural networks: Teaching old mlps new tricks via distillation. arXiv preprint arXiv:2110.08727. 2021 Oct 17.

97. Bai P, Miljković F, John B, Lu H. Interpretable bilinear attention network with domain adaptation improves drug–target prediction. Nat Mach Intell. 2023 Feb 1;5(2):126–36.

98. Li, M. et al. DGL-LifeSci: an open-source toolkit for deep learning on graphs in life science. ACS Omega 6, 27233–27238 (2021).

99. Zhan LM, Liu B, Fan L, Chen J, Wu XM. Medical Visual Question Answering via Conditional Reasoning. In: Proceedings of the 28th ACM International Conference on Multimedia. New York, NY, USA: ACM; 2020. p. 2345–54.

100. Long M, Cao Z, Wang J, Jordan MI. Conditional Adversarial Domain Adaptation.

101. Fontana A, De Laureto PP, Spolaore B, Frare E, Picotti P, Zambonin M. Probing protein structure by limited proteolysis. Acta Biochim Pol. 2004 Jun 30;51(2):299–321.

102. Gilson MK, Liu T, Baitaluk M, Nicola G, Hwang L, Chong J. BindingDB in 2015: A public database for medicinal chemistry, computational chemistry and systems pharmacology. Nucleic Acids Res. 2016;44(D1):D1045–53.

103. Zitnik, M., Sosič, R., Maheshwari, S. & Leskovec, J. BioSNAP datasets: Stanford biomedical network dataset collection. https:// snap.stanford.edu/biodata (2018).

104. Xu-Ying Liu, Jianxin Wu, Zhi-Hua Zhou. Exploratory Undersampling for Class-Imbalance Learning. IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics). 2009 Apr;39(2):539–50.

105. Paszke, A. et al. PyTorch: an imperative style, high-performance deep learning library. In Advances in Neural Information Processing Systems (NeurIPS, 2019).

106. Wang, M. et al. Deep graph library: a graph-centric, highly-performant package for graph neural networks. Preprint at arXiv https://arxiv.org/abs/1909.01315 (2019).

107. Harris, C. R. et al. Array programming with numpy. Nature 585, 357–362 (2020).

108. The pandas development team. pandas-dev/pandas: Pandas 1.2.4. Zenodo https://doi.org/10.5281/zenodo.4681666 (2021).

109. Ho, T. K. Random decision forests. In Int. Conf. on Document Analysis and Recognition, vol. 1, 278–282 (1995).

110. Sandhaus S, Annamalai T, Welmaker G, others. Small-Molecule Inhibitors Targeting Topoisomerase I as Novel Antituberculosis Agents. Antimicrob Agents Chemother. 2016;60:4028–36.

111. Teo JWP, Thayalan P, Beer D, others. Peptide Deformylase Inhibitors as Potent Antimycobacterial Agents. Antimicrob Agents Chemother. 2006;50:3665–73.

112. Szafran MJ, Kołodziej M, Skut P, others. Amsacrine Derivatives Selectively Inhibit Mycobacterial Topoisomerase I (TopA), Impair M. smegmatis Growth and Disturb Chromosome Replication. Front Microbiol. 9.

113. Quigley J, Peoples A, Sarybaeva A, others. Novel Antimicrobials from Uncultured Bacteria Acting against Mycobacterium tuberculosis. mBio. 11.

114. Palencia A, Li X, Bu W, others. Discovery of Novel Oral Protein Synthesis Inhibitors of Mycobacterium tuberculosis That Target Leucyl-tRNA Synthetase. Antimicrob Agents Chemother. 2016;60:6271–80.

115. Lamichhane G, Zignol M, Blades NJ, others. A postgenomic method for predicting essential genes at subsaturation levels of mutagenesis: application to Mycobacterium tuberculosis. Proc Natl Acad Sci U S A. 2003;100:7213–8.

116. Brown AK, Papaemmanouil A, Bhowruth V, others. Flavonoid inhibitors as novel antimycobacterial agents targeting Rv0636, a putative dehydratase enzyme involved in Mycobacterium tuberculosis fatty acid synthase II. Microbiology (Reading). 2007;153:3314–22.

117. Stanley SA, Kawate T, Iwase N, others. Diarylcoumarins inhibit mycolic acid biosynthesis and kill Mycobacterium tuberculosis by targeting FadD32. Proceedings of the National Academy of Sciences. 2013;110:11565–70.

118. Gurcha SS, Usha V, Cox JAG, others. Biochemical and Structural Characterization of Mycobacterial Aspartyl-tRNA Synthetase AspS, a Promising TB Drug Target. PLoS One. 9.

119. Buchieri MV, Riafrecha LE, Rodríguez OM, others. Inhibition of the β-carbonic anhydrases from Mycobacterium tuberculosis with C-cinnamoyl glycosides: Identification of the first inhibitor with anti-mycobacterial activity. Bioorg Med Chem Lett. 2013;23:740–3.

120. Murugesan D, Ray P, Bayliss T, others. 2-Mercapto-Quinazolinones as Inhibitors of Type II NADH Dehydrogenase and Mycobacterium tuberculosis: Structure–Activity Relationships, Mechanism of Action and Absorption, Distribution, Metabolism, and Excretion Characterization. ACS Infect Dis. 2018;4:954–69.

121. Jeankumar VU, Renuka J, Santosh P, others. Thiazole–aminopiperidine hybrid analogues: Design and synthesis of novel Mycobacterium tuberculosis GyrB inhibitors. Eur J Med Chem. 2013;70:143–53.

122. Krishna VS, Zheng S, Rekha EM, Guddat LW, Sriram D. Discovery and evaluation of novel Mycobacterium tuberculosis ketol-acid reductoisomerase inhibitors as therapeutic drug leads. J Comput Aided Mol Des. 2019 Mar 21;33(3):357–66.

123. Zhong W, Pasunooti KK, Balamkundu S, Wong YH, Nah Q, Gadi V, et al. Thienopyrimidinone Derivatives That Inhibit Bacterial tRNA (Guanine37- N^1)-Methyltransferase (TrmD) by Restructuring the Active Site with a Tyrosine-Flipping Mechanism. J Med Chem. 2019 Sep 12;62(17):7788–805.

124. San Jose G, Jackson ER, Uh E, Johny C, Haymond A, Lundberg L, et al. Design of potential bisubstrate inhibitors against Mycobacterium tuberculosis (Mtb) 1-deoxy-d-xylulose 5-phosphate reductoisomerase (Dxr)—evidence of a novel binding mode. Medchemcomm. 2013;4(7):1099.

125. Whitehouse AJ, Thomas SE, Brown KP, Fanourakis A, Chan DSH, Libardo MDJ, et al. Development of Inhibitors against *Mycobacterium abscessus* tRNA (m^1 G37) Methyltransferase (TrmD) Using Fragment-Based Approaches. J Med Chem. 2019 Aug 8;62(15):7210–32.

126. Gokhale KM, Telvekar VN. Novel peptidomimetic peptide deformylase (PDF) inhibitors of *Mycobacterium tuberculosis*. Chem Biol Drug Des. 2021 Jan 24;97(1):148–56.

127. Kasbekar M, Fischer G, Mott BT, Yasgar A, Hyvönen M, Boshoff HIM, et al. Selective small molecule inhibitor of the *Mycobacterium tuberculosis* fumarate hydratase reveals an allosteric regulatory site. Proceedings of the National Academy of Sciences. 2016 Jul 5;113(27):7503–8.

128. Bryk R, Gold B, Venugopal A, Singh J, Samy R, Pupek K, et al. Selective Killing of Nonreplicating Mycobacteria. Cell Host Microbe. 2008 Mar;3(3):137–45.

129. Yamaguchi T, Yokoyama K, Nakajima C, Suzuki Y. DC-159a Shows Inhibitory Activity against DNA Gyrases of Mycobacterium leprae. PLoS Negl Trop Dis. 2016 Sep 28;10(9):e0005013.

130. Aspatwar A, Hammarén M, Koskinen S, Luukinen B, Barker H, Carta F, et al. β-CA-specific inhibitor dithiocarbamate Fc14–584B: a novel antimycobacterial agent with potential to treat drug-resistant tuberculosis. J Enzyme Inhib Med Chem. 2017 Jan 1;32(1):832–40.

131. Soto R, Perez-Herran E, Rodriguez B, Duma BM, Cacho-Izquierdo M, Mendoza-Losana A, et al. Identification and characterization of aspartyl-tRNA synthetase inhibitors against Mycobacterium tuberculosis by an integrated whole-cell target-based approach. Sci Rep. 2018 Aug 23;8(1):12664.

132. Dkhar HK, Gopalsamy A, Loharch S, Kaur A, Bhutani I, Saminathan K, et al. Discovery of Mycobacterium tuberculosis α-1,4-Glucan Branching Enzyme (GlgB) Inhibitors by Structure- and Ligand-based Virtual Screening. Journal of Biological Chemistry. 2015 Jan;290(1):76–89.

133. Nisa S, Blokpoel MCJ, Robertson BD, Tyndall JDA, Lun S, Bishai WR, et al. Targeting the chromosome partitioning protein ParA in tuberculosis drug discovery. Journal of Antimicrobial Chemotherapy. 2010 Nov 1;65(11):2347–58.

134. Usha V, Hobrath J V., Gurcha SS, Reynolds RC, Besra GS. Identification of Novel Mt-Guab2 Inhibitor Series Active against M. tuberculosis. PLoS One. 2012 Mar 29;7(3):e33886.

135. Luckner SR, Machutta CA, Tonge PJ, Kisker C. Crystal Structures of Mycobacterium tuberculosis KasA Show Mode of Action within Cell Wall Biosynthesis and its Inhibition by Thiolactomycin. Structure. 2009 Jul;17(7):1004–13.

136. Mugumbate G, Abrahams KA, Cox JAG, Papadatos G, van Westen G, Lelièvre J, et al. Mycobacterial Dihydrofolate Reductase Inhibitors Identified Using Chemogenomic Methods and In Vitro Validation. PLoS One. 2015 Mar 23;10(3):e0121492.

137. Singh AK, Carette X, Potluri LP, Sharp JD, Xu R, Prisic S, et al. Investigating essential gene function in *Mycobacterium tuberculosis* using an efficient CRISPR interference system. Nucleic Acids Res. 2016 Oct 14;44(18):e143–e143.

138. Locher CP, Jones SM, Hanzelka BL, Perola E, Shoen CM, Cynamon MH, et al. A Novel Inhibitor of Gyrase B Is a Potent Drug Candidate for Treatment of Tuberculosis and Nontuberculosis Mycobacterial Infections. Antimicrob Agents Chemother. 2015 Mar;59(3):1455–65.

139. Makafe GG, Hussain M, Surineni G, Tan Y, Wong NK, Julius M, et al. Quinoline Derivatives Kill Mycobacterium tuberculosis by Activating Glutamate Kinase. Cell Chem Biol. 2019 Aug;26(8):1187-1194.e5.

140. Wescott HH, Zuniga ES, Bajpai A, Trujillo C, Ehrt S, Schnappinger D, et al. Identification of Enolase as the Target of 2-Aminothiazoles in Mycobacterium tuberculosis. Front Microbiol. 2018 Oct 26;9.

141. García MT, Carreño D, Tirado-Vélez JM, Ferrándiz MJ, Rodrigues L, Gracia B, et al. Boldine-Derived Alkaloids Inhibit the Activity of DNA Topoisomerase I and Growth of Mycobacterium tuberculosis. Front Microbiol. 2018 Jul 24;9.

142. Chen C, Han X, Yan Q, Wang C, Jia L, Taj A, et al. The Inhibitory Effect of GlmU Acetyltransferase Inhibitor TPSA on Mycobacterium tuberculosis May Be Affected Due to Its Methylation by Methyltransferase Rv0560c. Front Cell Infect Microbiol. 2019 Jul 17;9.

143. Abrahams KA, Chung C wa, Ghidelli-Disse S, Rullas J, Rebollo-López MJ, Gurcha SS, et al. Identification of KasA as the cellular target of an anti-tubercular scaffold. Nat Commun. 2016 Sep 1;7(1):12581.