Distribution Agreement

In presenting this thesis or dissertation as a partial fulfillment of the requirements for an advanced degree from Emory University, I hereby grant to Emory University and its agents the non-exclusive license to archive, make accessible, and display my thesis or dissertation in whole or in part in all forms of media, now or hereafter known, including display on the world wide web. I understand that I may select some access restrictions as part of the online submission of this thesis or dissertation. I retain all ownership rights to the copyright of the thesis or dissertation. I also retain the right to use in future works (such as articles or books) all or part of this thesis or dissertation.

Signature:

__________________________________________  _____________

Bonggun Shin  Date
Deep Learning Approaches Towards Computerized Drug Discovery

By

Bonggun Shin
Doctor of Philosophy

Computer Science and Informatics

Joyce C. Ho, Ph.D.
Advisor

Li Xiong, Ph.D.
Committee Member

Zhaohui S. Qin, Ph.D.
Committee Member

Jimeng Sun, Ph.D.
Committee Member

Accepted:

Lisa A. Tedesco, Ph.D.
Dean of the James T. Laney School of Graduate Studies

Date
Deep Learning Approaches Towards Computerized Drug Discovery

By

Bonggun Shin
B.A., Illinois Institute of Technology, IL, 2006
M.Sc., Korea Advanced Institute of Science and Technology, South Korea, 2010
M.Sc., Emory University, GA, 2019

Advisor: Joyce C. Ho, Ph.D.

An abstract of
A dissertation submitted to the Faculty of the
James T. Laney School of Graduate Studies of Emory University
in partial fulfillment of the requirements for the degree of
Doctor of Philosophy
in Computer Science and Informatics
2020
Proposing a new drug candidate is an essential part of the drug discovery process, consisting of many sub-tasks. Traditionally, these tasks have been tackled by chemistry and pharmaceutical experts and take years to design. Therefore, this thesis aims to accelerate drug discovery by proposing deep-learning models that accomplishes these tasks effectively and quickly. For the target identification problem, we propose new feature selection methods for both disease-related and prognosis-related features. Next, we propose a new drug-target interaction model to perform the drug re-purposing task. In this model, we present a new molecule representation to overcome the limitation of the current models. We also propose a novel drug candidate generation model that can modify an existing drug to meet given molecule properties. For each project, we present an empirical evaluation to show the competency of the proposed approaches. In addition, we also provide analyses or case studies to demonstrate the practicality of our approaches.
Deep Learning Approaches Towards Computerized Drug Discovery

By

Bonggun Shin
B.A., Illinois Institute of Technology, IL, 2006
M.Sc., Korea Advanced Institute of Science and Technology, South Korea, 2010
M.Sc., Emory University, GA, 2019

Advisor: Joyce C. Ho, Ph.D.

A dissertation submitted to the Faculty of the
James T. Laney School of Graduate Studies of Emory University
in partial fulfillment of the requirements for the degree of
Doctor of Philosophy
in Computer Science and Informatics
2020
This work represents the realization of my lifelong goal, and the way to it has never been smooth. Thankfully, I have been guided and supported by many remarkable people, and without them, this work could not be realized as it is now. Although my appreciation that I hold for them cannot be fully expressed in just a short text, I’d like to remember them throughout my life by frequently reminding these brief acknowledgments.

I am profoundly grateful to my advisor Prof. Joyce Ho for her mentorship, encouragement, and supports. She encouraged me to explore new research areas, gave me insights, and waited for me so that I can overcome hardships by myself. I am extremely blessed to have been a graduate student of such a wonderful academic mentor.

I would like to extend my thanks to the members of my dissertation committee: Prof. Lee Xiong, Prof. Steve Qin, and Prof. Jimeng Sun. for helping me to improve the quality of my dissertation through valuable comments and feedbacks.

Besides the committee members, I would like to also thank Prof. Jinho Choi. For the earlier years of Emory, he trained me to be a better researcher. Thanks to his training, my programming and academic writing have been dramatically improved.

I also want to express my gratitude to the excellent collaborators, all members of Deargen. They have introduced me to interesting real-world problems and supported me to solve them together. The majority parts of this dissertation would not be possible without this collaboration.

I would like to give a special thanks to my soul mate, Kim Heonoo and Sunho
Hwang. I’ll remember your prayers, supports, and love especially in the midst of hardships.

Finally, this dissertation is dedicated to my parents, Seunghae Shin and Yongsoon Kim; my sister, Eunkyung Shin, my brothers, Seokki Kwon and Keumku Kang; my son, Caleb Taeha Shin, and lastly but most importantly my wife, Junghyun Kang. Your unwavering love has been the motivation of my life and this work. I’m the most fortunate one in the world because I met all of you. Thank you and I love you all.

– Bonggun Shin

*Emory University*

*March 2020*
## Contents

1 Introduction  
   1.1 Drug Discovery Pipeline  
   1.2 AI in Drug Discovery  
   1.3 Contributions  
   1.4 Outline  

2 Disease-Related Target Identification  
   2.1 Motivation  
   2.2 Problem Definition  
   2.3 Proposed Model: Wx  
   2.4 Experiments  
   2.5 Discussion  
   2.6 Contribution  

3 Prognosis-Related Target Identification  
   3.1 Motivation  
   3.2 Problem Definition  
   3.3 Proposed Model: Cascaded Wx  
   3.4 Experiments  
   3.5 Discussion  
   3.6 Contribution
# 4 Drug Repurposing

4.1 Introduction .................................................. 37
4.2 Related Work .................................................. 41
4.3 Problem Definition .............................................. 42
4.4 Proposed Model: Molecule Transformer-Drug Target Interaction ... 42
  4.4.1 Model Architecture ........................................ 43
  4.4.2 Molecule Transformers ..................................... 44
  4.4.3 Protein CNNs ................................................ 49
  4.4.4 Interaction Denses .......................................... 49
4.5 Experiments .................................................... 50
  4.5.1 Datasets .................................................... 50
  4.5.2 Training Details .......................................... 51
  4.5.3 Evaluation Metrics ........................................ 53
  4.5.4 Baselines .................................................. 54
  4.5.5 Results .................................................... 55
4.6 Case Studies .................................................... 56
  4.6.1 Anticancer Drug Discovery ................................. 56
  4.6.2 Antiviral Drug Discovery .................................. 58
4.7 Discussion ...................................................... 63
4.8 Contribution .................................................... 64

# 5 Molecule Generation

5.1 Introduction .................................................... 65
5.2 Related Work .................................................... 68
5.3 Problem Definition .............................................. 70
5.4 Proposed Model: Controlled Molecule Generator .................. 70
  5.4.1 Model Overview ............................................ 71
  5.4.2 Background ................................................ 71
5.4.3 Molecule Translation Network .................................. 73
5.4.4 Constraint Networks .............................................. 75
5.4.5 Modified Beam Search with Constraint Networks .......... 79
5.4.6 Diversifying the Output ........................................... 81
5.5 Experiments ............................................................. 81
  5.5.1 Datasets ............................................................ 82
  5.5.2 Pre-Training of Constraint Networks ....................... 84
  5.5.3 Single Objective Optimization ................................. 84
  5.5.4 Multi Objective Optimization ................................. 87
  5.5.5 Ablation Study .................................................... 90
5.6 Case Study ............................................................. 91
5.7 Discussion ............................................................. 92
5.8 Contribution ........................................................... 93

6 Conclusion ............................................................... 94

7 Future Direction ......................................................... 96

Bibliography ............................................................... 97
List of Figures

1.1 Drug discovery pipeline and the sub-tasks in the Drug Cadidates phase. 2

2.1 The conventional biomarker discovery pipeline using assorted machine learning methods. Figure is from [120]. 10

2.2 Training step of Wx method. We train the given network using the datasets consisting of genes as features and disease labels as the truth values of outputs. 13

2.3 Vector extraction step of Wx method. We extract weights \((W_N, W_C)\) and averaged input vectors \((\hat{X}_N, \hat{X}_C)\) from the trained model. 14

2.4 Discriminating power analysis. 15

2.5 Performance of the Wx-14-UGCB on BRCA, LUAD, and LUSC RNA-seq data. AUC values are listed. ROC, receiver operating characteristic. 19

2.6 Classification accuracy according to given number of genes. The x-axis indicates the number of top genes (sorted in descending order by the DI values) used for the calculation and the y-axis represents the average accuracy. 20

2.7 Comparison of genes identified by Wx and edgeR. The x-axis indicates the number of top genes used for the comparison and the y-axis represents the percentage of overlap between the gene sets. 20

2.8 Key networks of the top 50 genes with highest DI scores are shown. 21
3.1 An example of CWx’s feature selection procedure. Input samples (patients) are reduced through three cascaded steps using different criteria. For each step, a different cutoff criteria is used in an easy to hard manner. For example, three-year, 2 versus 4-year, and 1 versus 5-year cutoffs for grouping samples into either high- or low-risk groups are used at the first, second, and third steps, respectively. Input features (genes) are also reduced by a quarter in each step. Finally, the prognostic potential of features can be estimated according to the weights calculated from the trained neural network.

3.2 Violin plot of the comparison of feature selection algorithms with top 100 genes. The metric is a c-index with the selected 100 genes in lung adenocarcinoma (LUAD) samples for each algorithm (left). White circles indicate the medians; box limits inside the polygons indicate the 25th and 75th percentiles as determined by R software; whiskers extend 1.5 times the interquartile range from the 25th and 75th percentiles; polygons represent density estimates of data and extend to extreme values. Asterisks (*p < 0.05 and **p < 0.001) indicate the results of one-way ANOVA (p < 0.0001) with post hoc test (pairwise t test with BonferroniHolm correction). x- and y-axes indicate the number of cumulative top genes and c-index, respectively (right).

3.3 Gene ontology (GO) analysis of top 100 genes. GO analysis was performed using Metascape (http://metascape.org/gp/index.html) with top 100 genes (default parameters were used). The significance of a given GO term is represented by gray (significant) or white (nonsignificant) bars with a P cutoff value of 0.0001.
4.1 The Proposed DTI Model Architecture. Inputs are molecule (SMILES) and protein (FASTA) and the regression output is the affinity score between these two inputs. .............................. 43
4.2 Three parts of the proposed model. .................................................. 44
4.3 An example of molecule token embedding (MTE) and positional embedding (PE) to make the model input $x_i$ for a given molecule sequence of methyl isocyanate (CN=C=O). .............................. 46
4.4 Put your caption here ................................................................. 47
5.1 The proposed controlled molecule generator model. ......................... 67
5.2 The molecule translation network. .................................................. 74
5.3 Two Constraint Networks. .............................................................. 76
5.4 A case study: The Aniracetam optimization task to improve DRD2 score. The molecule produced by the proposed model achieved the better DRD2 score than the molecule optimized by MolDQN$^1$. ........ 91
List of Tables

1.1 Caption for LOF ............................................. 1

2.1 The number of cancer and normal samples used in this study. .... 18

2.2 Classification accuracy comparison in different cancer types of TCGA datasets(%) ............................................. 18

2.3 The classification accuracy of the UGCBs for non TCGA dataset (%). 21

2.4 Gene expression biomarkers identified by different studies. ......... 23

2.5 Genes ranked by the discriminative index score. * indicates genes known as house keeping genes. ............................................. 24

2.6 Classification accuracy (%) of non-cancer transcriptomic data set. .. 25

4.1 Statistics of the Davis and Kiba datasets. TRN/DEV/TST: training, development, evaluation sets. ............................................. 50

4.2 Test set results of the proposed MT-DTI model, MT-DTI model without fine-tuning (denoted as MT-DTI$^{w/oFT}$), and other existing approaches. ............................................. 55

4.3 Compound ranking based on the predicted Kiba scores when the target is EGFR protein. All compounds are from Drugbank database excluded any compounds in Kiba dataset. [Compound Name]* represents a known EGFR targeting drug. ............................................. 57
4.4 DTI prediction results of antiviral drugs available on markets against a COVID-2019 targeting 3C-like proteinase. Ritonavir is expressed in canonical and isomeric SMILES, and * indicates the isomeric SMILES of ritonavir.

4.5 DTI prediction results of antiviral drugs available on markets against a COVID-2019 targeting RNA polymerase.

4.6 DTI prediction results of antiviral drugs available on markets against a COVID-2019 targeting helicase.

4.7 DTI) prediction results of antiviral drugs available on markets against COVID-2019 targeting 3’-to-5’ exonuclease.

4.8 DTI prediction results of antiviral drugs available on markets against a COVID-2019 targeting endoRNAse.

4.9 DTI prediction results of antiviral drugs available on markets against a COVID-2019 targeting 2’-O-ribose methyltransferase.

5.1 Molecules used in this study. About 54% of DRD2D molecules also belong to ZINC, therefore, the total number of molecules for this study is 260,939.

5.2 Single objective optimization performance comparison on the penalized logP task. MolDQN results are from, and the scores of other baselines are from.

5.3 Multi objective optimization performance comparison.

5.4 Ablation study. MBS is modified beam search, PNet is PropNet, and SNet is SimNet.
# List of Algorithms

1. Discriminative Index .................................................. 16
2. Modified Beam Search .................................................. 80
1

Introduction

*If even one new drug of the stature of penicillin or digitalis has been unjustifiably banished to a company’s back shelf because of exceedingly stringent regulatory requirements, that event will have harmed more people than all the toxicity that has occurred in the history of modern drug development.*

– William Wardell

Drug discovery is a challenging, time-consuming and costly process, which takes an average of 9 to 12 years to develop a single drug [42]. Any failure in this costly and lengthy process can lead to enormous financial losses. What’s worse is failures occur quite often throughout the drug development pipeline. According to [43], the estimated average cost to develop a new medicine and gain FDA approval is $1.4 billion. Among this amount, 40% of it is spent on the candidate compound genera-

<table>
<thead>
<tr>
<th></th>
<th>Candidates</th>
<th>Pre-clinical</th>
<th>Phase 1</th>
<th>Phase 2</th>
<th>Phase 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time</td>
<td>4-5 years</td>
<td>1-2 years</td>
<td>1-2 years</td>
<td>1-2 years</td>
<td>2-3 years</td>
</tr>
<tr>
<td>Cost</td>
<td>$550M</td>
<td>$125M</td>
<td>$225M</td>
<td>$250M</td>
<td>$250M</td>
</tr>
<tr>
<td>Molecules</td>
<td>5k-10k</td>
<td>10-20</td>
<td>5-10</td>
<td>2-5</td>
<td>1-2</td>
</tr>
</tbody>
</table>

Table 1.1: Traditional Drug Discovery and Development Process*. It consists of drug candidate generation step (Candidates), cell-line and animal experiment step (Pre-clinical), and three phase of human clinical tests (Phase 1,2, and 3).

* The table is adapted from the slide of Data Mining for Drug Discovery in KDD19.
tion step as summarized in Table 1.1. In this step, around 5,000 to 10,000 molecules are generated as candidates but over 99.9% of them will be eventually discarded and only 0.1% of them will be approved to the market. This low approval rate is also attributed to the stringent regulations to protect people from unknown adversarial effects. However, as Wardell pointed out in the quote, it’s important to continue to develop new drugs despite these difficulties. One of the efforts to overcome these challenges is adopting computer-aided systems in many subtasks of the drug discovery pipeline. With the help of intelligent and automated systems, subtasks of the drug discovery process can be optimized to make the entire process cost-effective. We first overview those subtasks in the pipeline (Section 1.1) and introduce the current trends of computer-aided drug discovery (Section 1.2). Inspired by the areas that have overcome the limitations and made many breakthroughs by applying deep learning, we discuss what innovations we have made in drug discovery through this work (Section 1.3).

1.1 Drug Discovery Pipeline

The drug discovery process involves several disciplines such as genomics, chemistry, biology, and pharmacology, which can be summarized as a series of five stages (Figure 1.1).

Figure 1.1: Drug discovery pipeline and the sub-tasks in the Drug Candidates phase.
• **Target Identification**: The goal of this stage is to find biological entities (proteins, genes or RNAs) associated with a specific disease with which a putative drug interacts [4]. Found targets need to be validated using its ability to regulate biological processes. Through numerous studies, we confirm the relationships between the target and the disease state [123].

• **Molecule Discovery**: Molecule discovery involves the finding of synthetic chemicals that shows a degree of efficacy for a given target and potentially aims to cure diseases associated with the target [82]. Molecules can be divided into two types based on its molecular weights: small and large molecules. In this work, we refer to small molecules as molecules because they account for about 70% of all drugs¹. When finding a molecule candidate, we can generate a new molecule (Section 5) or repurpose a known molecule to a new target (Section 4).

• **Molecule Optimization**: At the molecule optimization stage, we improve the potency and other important properties of a candidate molecule. The goal of the optimization is to prioritize and select promising candidates with safety and potency.

• **Pre-clinical Test**: These selected candidates are tested on animals for potency and toxicity before tested on human volunteers. Using in vivo animal studies, we also characterize toxicity profiles for various doses [147, 19].

• **Clinical Test**: The clinical test consists of three phases that investigate side-effects, safety, dosage profiles, potency and other properties of the candidate molecule on human volunteers [147]. If any drug passed all three stages of the clinical test, then it is approved by FDA and available to the market.

The advancement of computing technology has led to the accumulation of large biological entity databases with its bioactivity profiles [117, 83]. As such, the con-

ventional drug discovery methodology, which relies on manual works of expert groups is reaching their limit, because the dataset is now too large for humans to extract valuable information from it. This inefficient nature of the candidate generation step serves as motivation to consider an alternative method, artificial intelligence (AI) based drug discovery. The advantage of AI platforms is higher expected success rate by reducing the processing time because it can systematically generate candidate molecules and analyze immense amounts of chemical and biological data in a short period of time.

### 1.2 AI in Drug Discovery

Recently, pharmaceutical giants have started to collaborate with small companies actively developing drug discovery software using AI. As examples, Takeda Pharmaceutical is using Numerates AI platform to find small-molecules for oncology, gastroenterology, and central nervous system disorders. Pfizer integrates IBM Watson for Drug Discovery into its pipeline to effectively utilize Pfizer’s scientific knowledge with a set of machine learning methods expecting the improved search for immunoncology drugs. Eli Lilly is supporting Atomwise to generate promising drug candidates using its deep learning-based molecule screening tool. One might be skeptical of this new approach because there have been few success stories, however, if shown to be effective, then this would become a new standard pipeline because it could save much time and cost. Not only that, just like many AIs have created innovation based on deep learning, new drug development methodologies can make a big breakthrough with deep learning.

In an image recognition task, for example, the performance of computer models was significantly behind the one of a human until 2010, when the ImageNet [39] project begun. In the ImageNet project, many people gathered, cleaned and annotated about
1.4 billion of images, with which many researchers have proposed advanced deep learning models. With the help of the well-curated dataset and appropriate computational power, many deep learning models can surpass human performance [65, 73, 129, 155]. Similar to computer vision, natural language process (NLP) has gone through a long dark period, however, since Google AI’s BERT [41] and Transformer [160] have been proposed, deep learning models have achieved another level. There are two main factors to this success; availability of high-quality large data sets and the rapid development of powerful computational hardware.

These success stories have motivated drug discovery researchers to adopt advanced deep learning methods to one of the sub-tasks in the drug discovery pipeline. As described in Section 1.1, there are many sub-tasks in the pipeline, many of which have not adopted the recently developed deep learning techniques or still relied on manual labors of experts. For example, the previous state of the art of the drug toxicity prediction problem [181] had been based on an ensemble of three traditional machine learning models with assorted features. Although it was accurate, it’s less practical because preparing all these different kinds of features and feeding them into multiple models are laborious. Recently, an end-to-end deep learning-based method [2] has shown the most accurate prediction performance in the toxicity prediction. The authors leveraged a large drug database, ZINC [74] to pre-train the network so that a modern deep learning model could be used to solve the problem. Therefore, if a deep learning model is carefully designed to effectively reflect the characteristics of the problem, we can make breakthroughs in drug discovery.

1.3 Contributions

With this motivation, this dissertation aims to design a novel deep learning based model for each of the selected sub-tasks in the drug discovery pipeline summarized in
the green boxes in Figure 1.1. In particular, we focus on the following three subtasks in this work:

- **Target identification**: We propose new feature selection methods for both disease-related and prognosis-related features. A biological target can be associated with a specific disease or a prognosis of a disease. Since the number of genes is about 20,000, we can pose the target identification problem as a feature selection task. Therefore, we introduce two feature selection algorithms with respect to disease detection and prognosis awareness.

  [Related publications]:

- **Molecule Discovery (Drug repurposing)**: We propose a new drug-target interaction model to perform the drug re-purposing task. In this model, we present a new molecule representation to overcome the limitation of the current models. In addition to experiments on benchmarks, we also present molecule candidates from commercially available antiviral drugs that may cure the novel coronavirus (COVID-19).

  [Related publications]:
  - **Bonggun Shin**, Sungsoo Park, Keunsoo Kang, and Joyce C. Ho, Self-

* indicates equal contribution
Attention Based Molecule Representation for Predicting Drug-Target Interaction, MLHC 2019 (30.9%)

- Bo Ram Beck, Bonggun Shin, Yoonjung Choi, Sungsoo Park, and Keunsoo Kang, Predicting commercially available antiviral drugs that may act on the novel coronavirus (COVID-19) through a drug-target interaction deep learning model, Computational and Structural Biotechnology Journal (IF=4.72)

- **Molecule Discovery (Optimized drug candidate generation):** We propose a novel drug candidate generation model that can modify an existing drug to meet given molecule properties. This model considers two subtasks at the same times: molecule generation and molecule optimization.

  [Related publication]:

  - Bonggun Shin, Sungsoo Park, and Joyce C. Ho, Controlled Molecule Generation via Self-Attention based Translation, Submitted to KDD (20%)

For each project, we present an empirical evaluation to show the competency of the proposed approaches. In addition, we also provide analyses or case studies to demonstrate the practicality of our approaches.

1.4 Outline

The rest of this dissertation is structured as three primary chapters. Chapter 2 and Chapter 3 introduce two feature selection methods one for disease-related and another for prognosis-related features. Both of them are based on neural networks and the latter is built upon the former. In Chapter 4, we present a new drug-target interaction model based on self-attention mechanism, which has shown its potential in the natural language process domain. Chapter 5 propose a new controlled molecule generating model which optimizes the input molecule to improve multiple properties in a single
end-to-end model. Finally, Chapter 6 presents the conclusion of the dissertation and discusses the future direction of the work.
2

Disease-Related Target Identification

The target identification projects primarily consist of two parts: the disease-related feature selection algorithm (Section 2) and the prognosis-related feature selection algorithm (Section 3). The two parts have been finished and published as first authored papers to Nature Scientific Report [118] and Frontiers in Genetics [142], respectively. Both of them, I was an equally contributed first author with Park.

2.1 Motivation

Gene (or protein) biomarkers clarify the health state of a patient and predict the potential response to a candidate drug as well [109]. The conventional discovery process of these biomarkers demands comprehensive requirements; a broad set of expert groups, such as biologists, statisticians, and clinicians and other experimental supports, including laboratory and complicated software tools along with another group of experts in these areas. The first stage of this process is based on the manual selection of genes or proteins with limited and biased information from the literature or experts. Then, experimental validation is followed to confirm the selected biomarker
different organs or different tissues in the same organ. Interestingly, three types of kidney tumors don’t show these patterns. We found that KIRC and KIRP are more similar to each other than KICH since they share 36% of DE genes (Table 1). Studies have shown that KICH is less aggressive than KIRC and KIRP.

Gene expression changes with phenotypic consequences are driven by mutations and epimutations. The driver mutations and epimutations may be scattered in different pathways. We hypothesize that some of these mutations or epimutations may disrupt a pathway responsible for cell cycle regulation that directly drives cells into uncontrolled proliferation, while others may lie within an organ-specific

Figure 2.1: The conventional biomarker discovery pipeline using assorted machine learning methods. Figure is from [120].

candidates. The selection of a promising biomarker is important and critical because an experimental validation is costly. To satisfy these requirements, many machine learning based biomarker selection methods have been proposed such as unsupervised clustering [126], gene ontology (GO) analysis [40], sparse regularization [185], differentially expression gene (DEG) analysis [98, 130], and a hybrid of existing methods (PENG) [120].

Among these methods, DEG is widely used for the identification of biomarker candidates. It narrows down to a succinct subset of genes from around 200,000 genes by looking at significantly altered expressions among different groups with a statistical threshold, an adjusted p-value of 0.05. Although it provides statistically meaningful genes, it becomes less practical recently because the number of selected genes tends to be increased to several thousand due to the increased number of raw features. The major reason for increased features is the reduced cost of sequencing. Consequently, it has become difficult for researchers to select the parsimonious sets of biomarker
candidates from a large number of DEG results. To mitigate this problem, several machine learning-based algorithms have been proposed [120, 132, 121]. In particular, [120] extracted 14-gene signatures from multiple types of cancers using a pipeline of various existing machine learning methods, as illustrated in Figure 2.1. These concise signature gene panels are valuable in the diagnosis and treatment of cancer. However, the complicated structure of this approach hinders it from widespread usage because it requires running each pipeline methods separately, which is laborious and time-consuming. To mitigate this problem, we aim to propose a new approach, called Wx, that can achieve both high predictive accuracy and simplicity for users.

Wx is our feature selection model for disease-related target identification model. The name Wx is the combination of a typical neural network weight (W) and input (x) as the feature importance is calculated based on weights and inputs together. Section 2.2 explains the formulation of the problem, and Section 2.3 details how we train the weights, and presents the feature selection algorithms, Wx.
2.2 Problem Definition

- **Data**: Gene data with disease diagnosis
  - \(X\): 20,000 different genes (features)
  - \(Y\): cancer/normal (binary classification)

- **Task**: Select top \(k\) features that contributes most in the classification task.

In this project, the goal is to select a concise subset of most important genes from the whole 20,000 genes, the result of RNA sequencing data. Although [120] successfully extracted 14 genes that show the promising predictive accuracy, it is not an end-to-end model, rather it relies on many third-party libraries, which makes it less accessible. Therefore, the proposed model should not only be easy to use, but select the most informative features.

Let \(X\) be \(N\) number of gene expressions for tumor or normal samples, then it can be formally expressed as \(X = \{X_1, X_2, ..., X_N\}\). Each \(X_i\) has \(J\) number of features \((X_i \in \mathbb{R}^J)\), each of which conveys information regarding the total expression amount of the corresponding gene. The output value \(Y \in \mathbb{R}^K\) is a one-hot vector that consists of \(K\) numbers depending on how many classes it represents. In formal notation, the vector \(Y\) can be expressed as \(Y = [y_1, y_2, \cdots, y_K]\). A binary class case, for example, is to classify tumor samples out of normal samples. Then the \(i\)-th input data with gene expression becomes \(X_i = [x_{i1}, x_{i2}, \cdots, x_{ij}]\), and the output becomes \(y_i\). If the \(i\)-th data is from a normal sample, then \(y_i = [1, 0]\), otherwise \(y_i = [0, 1]\).
2.3 Proposed Model: Wx

The proposed approach consists of three steps, training the network, extracting useful vectors, and discriminating power analysis. Throughout these steps, we use a feed-forward neural network with the softmax activation [119]. The reason why we use softmax is that the dependent variable of this task is categorical. When illustrating the idea, we assume the number of class is two for the simplicity, although the number of categories can be arbitrary as presented in Algorithm 1.

![Figure 2.2: Training step of Wx method. We train the given network using the datasets consisting of genes as features and disease labels as the truth values of outputs.](image)

**[Step 1] Training the network:** Figure 2.2 shows how to train the given network using the gene datasets. We denote each data sample as \(X_i\) and each corresponding annotation as either normal (N) or cancerous (C). The model is an one-layered dense network, therefore, we can denote the weights as \(W_N\) and \(W_C\), each of these is a vector with the dimension is equal to the number of features. In general, we can express this network with the following equation:
This softmax classification network includes the model parameters $W = \{W_1, W_2, ..., W_K\}$ that are learned from the training data. With these parameters, the prediction of the output, $\hat{Y}_i$, can be expressed as Equation 2.1 along with the input $X_i$. This network is trained using the dataset (cancerous and normal samples as shown in Figure 2.2).

**Figure 2.3**: Vector extraction step of Wx method. We extract weights $(W_N, W_C)$ and averaged input vectors $(\hat{X}_N, \hat{X}_C)$ from the trained model.

**[Step 2] Extracting useful vectors**: With the trained model, we extract two kinds of vectors to calculate the feature importance. The first group of vectors is the trained parameters, $W$. For the binary case example in Figure 2.3, the parameter vectors are $W_N$ and $W_C$, the weights of normal class and the weights of cancer class, correspondingly. Another group of vectors is the averaged input vectors denoted as
\( \hat{X}_N \) and \( \hat{X}_C \) in Figure 2.3, which can be expressed in the following equation.

\[
\hat{X}_k = \frac{1}{N_k} \sum_{X_i \in \text{Class } k} X_i
\]  

(2.2)

The summation term in Equation 2.2, \( X_i \in \text{Class } k \), represents all \( X_i \)'s where \( Y_i = k \). Each averaged input vector is the element-wise mean of all input vectors that belong to the class of interest. For the binary example in Figure 2.3, \( \hat{X}_N \) is calculated from all normal samples, and \( \hat{X}_C \) is calculated from all cancer samples.

[Step 3] **Discriminating power analysis**: For each class, we calculate an element-wise product (Hadamard product) between the corresponding weight vector and the averaged input vector as follows:

\[
WX_k = W_k \odot \hat{X}_k
\]  

(2.3)

Then, we get the absolute difference from all pairs of \( WX_k \) and \( WX_j \), which is the discriminating index (DI), the output of the method. Using this index, we can select top-k important features. For the binary example in Figure 2.3, DI is calculated from one absolute difference term from \( WX_C \) and \( WX_N \).
Algorithm 1 generalizes all these steps for a general case, where the number of classes is greater than 2. The network parameters, $W$, and the dataset, $X$ and $Y$, and the desired number of selected features, $c$, serves as the input to Discriminative Index algorithm (Algorithm 1).

**Algorithm 1: Discriminative Index**

**input**: $X, Y, \Theta, c$

**output**: $c$ number of gene names

1. Let $X^k$ be the input vector with class label $k$;
2. for $k \leftarrow 1$ to $K$
   3. $\hat{X}^k \leftarrow$ average($X^k$);
   4. $WX_k \leftarrow \theta_k \odot \hat{X}^k$;
5. end
6. for $j \leftarrow 1$ to $J$
7.   $DI_j \leftarrow 0$ ;
8.   foreach combination pair $(a, b)$ in $\{1, 2, \cdots, K\}$ do
9.     $DI_j \leftarrow DI_j + |WX_a(j) - WX_b(j)|$
10. end
11. end
12. $DI_{sort} \leftarrow$ Sorted $DI$ in descending order;
13. Return top $c$ gene names in $DI_{sort}$;

The commentary of this algorithm is as follows.

- Classify $X$ into $K$ classes according to its corresponding $Y$ which is denoted as $X^1, X^2, \ldots, X^K$.
- For each $X^k$, take average for all instances to form an average vector, $\hat{X}^k \in \mathbb{R}^J$
- Calculate the hadamard product between the parameter related to the k’th softmax output value, $\theta^k$ and the average vector, $\hat{X}^k$, which is assigned to $WX_k$.
- Aggregate the element wise differences between all combination pairs of $WX_k$ to get the discriminant index, $DI \in \mathbb{R}^J$. The example with $K = 3$ is illustrated in Figure 2.4.
• After the iteration (Line 6-11 in Algorithm 1), the resulting $DI$ is a vector of size $J$. This vector is sorted to form the sorted index, $DI_{sort}$.

• The final $c$ features (genes) are the indices of the top $c$ indices in $DI_{sort}$.

2.4 Experiments

In the experiments, we use gene expression data (mRNASeq) of 12 different cancer types from the cancer genome atlas (TCGA). Each sample contains normalized expression levels of 20,501 genes (features). A description of the TCGA data can be found in Table 2.1. We also used other RNA-seq datasets for validation of the proposed method; GSE720568 contains normalized expression levels of 23,686 genes performed in 1,257 malignant and 3,256 benign samples. GSE4041921 consists of normalized expression levels of 36,741 genes performed in 164 samples (87 lung cancer and 77 adjacent normal tissues). GSE103322 contains normalized expression levels of 23,686 genes, performed in 5,578 head and neck squamous cell carcinoma single cells (2,215 cancer cells and 3,363 non-cancer cells).

We choose the classification model as XGBoost [23] because it’s known as one of the best classification models. We excluded each set of training samples when validating the classification performance. For a fair comparison, we selected top 14 features among 20,501 like [120] did in their research. In formal notation, these new inputs can be represented as $X_{\text{selected}}^{v} \in \mathbb{R}^{N \times c}$. Since this dataset is imbalanced and data hungry, we use Leave-one-out cross validation (LOOCV), where only one sample is set aside for the test and the rest of them are used as the train set.

The first part of the experiments is on TCGA datasets. As summarized in Table 2.2, we compared our 14 features (Wx-14-UGCB) against Peng’s 14 [120] (Peng-14-UGCB) and edgeR [130] for 12 different cancer types. Wx-14-UGCB outperformed others on average by about 1.8%p. Specifically, out of 12 cancer types, Wx-14-UGCB
<table>
<thead>
<tr>
<th>Type ID</th>
<th>Full name</th>
<th># of cancer samples</th>
<th># of normal samples</th>
<th># of total samples</th>
<th>Cancer Ratio</th>
</tr>
</thead>
<tbody>
<tr>
<td>BLCA</td>
<td>Bladder urothelial carcinoma</td>
<td>408</td>
<td>19</td>
<td>427</td>
<td>0.95</td>
</tr>
<tr>
<td>BRCA</td>
<td>Breast invasive carcinoma</td>
<td>1101</td>
<td>113</td>
<td>1214</td>
<td>0.90</td>
</tr>
<tr>
<td>COAD</td>
<td>Colon adenocarcinoma</td>
<td>286</td>
<td>41</td>
<td>327</td>
<td>0.87</td>
</tr>
<tr>
<td>HNSC</td>
<td>Head and neck squamous cell carcinoma</td>
<td>522</td>
<td>44</td>
<td>566</td>
<td>0.92</td>
</tr>
<tr>
<td>KICH</td>
<td>Kidney chromophobe</td>
<td>65</td>
<td>25</td>
<td>90</td>
<td>0.72</td>
</tr>
<tr>
<td>KIRC</td>
<td>Kidney renal clear cell carcinoma</td>
<td>534</td>
<td>72</td>
<td>606</td>
<td>0.88</td>
</tr>
<tr>
<td>KIRP</td>
<td>Kidney renal papillary cell carcinoma</td>
<td>291</td>
<td>32</td>
<td>323</td>
<td>0.90</td>
</tr>
<tr>
<td>LIHC</td>
<td>Liver hepatocellular carcinoma</td>
<td>374</td>
<td>50</td>
<td>424</td>
<td>0.88</td>
</tr>
<tr>
<td>LUAD</td>
<td>Lung adenocarcinoma</td>
<td>517</td>
<td>59</td>
<td>576</td>
<td>0.89</td>
</tr>
<tr>
<td>LUSC</td>
<td>Lung squamous cell carcinoma</td>
<td>502</td>
<td>51</td>
<td>553</td>
<td>0.90</td>
</tr>
<tr>
<td>PRAD</td>
<td>Prostate adenocarcinoma</td>
<td>497</td>
<td>52</td>
<td>549</td>
<td>0.90</td>
</tr>
<tr>
<td>THCA</td>
<td>Thyroid carcinoma</td>
<td>512</td>
<td>59</td>
<td>571</td>
<td>0.89</td>
</tr>
</tbody>
</table>

Table 2.1: The number of cancer and normal samples used in this study.

<table>
<thead>
<tr>
<th>Model</th>
<th>Wx</th>
<th>Peng’s</th>
<th>edgeR</th>
</tr>
</thead>
<tbody>
<tr>
<td>BLCA</td>
<td>95.79</td>
<td><strong>97.20</strong></td>
<td>94.86</td>
</tr>
<tr>
<td>BRCA</td>
<td><strong>98.19</strong></td>
<td>96.38</td>
<td>91.78</td>
</tr>
<tr>
<td>COAD</td>
<td>94.51</td>
<td>87.20</td>
<td><strong>98.78</strong></td>
</tr>
<tr>
<td>HNSC</td>
<td><strong>97.17</strong></td>
<td>92.23</td>
<td>94.35</td>
</tr>
<tr>
<td>KICH</td>
<td>95.65</td>
<td>95.65</td>
<td><strong>100.00</strong></td>
</tr>
<tr>
<td>KIRC</td>
<td><strong>99.67</strong></td>
<td>96.70</td>
<td>99.34</td>
</tr>
<tr>
<td>KIRP</td>
<td><strong>99.38</strong></td>
<td>97.53</td>
<td><strong>99.38</strong></td>
</tr>
<tr>
<td>LIHC</td>
<td>90.57</td>
<td><strong>94.81</strong></td>
<td>87.74</td>
</tr>
<tr>
<td>LUAD</td>
<td>97.92</td>
<td>97.58</td>
<td><strong>98.96</strong></td>
</tr>
<tr>
<td>LUSC</td>
<td>98.19</td>
<td>96.75</td>
<td><strong>99.28</strong></td>
</tr>
<tr>
<td>PRAD</td>
<td><strong>93.45</strong></td>
<td>94.55</td>
<td>92.36</td>
</tr>
<tr>
<td>THCA</td>
<td><strong>95.80</strong></td>
<td>89.86</td>
<td>90.21</td>
</tr>
</tbody>
</table>

Table 2.2: Classification accuracy comparison in different cancer types of TCGA datasets(%).
Our comparison revealed that UGCBs identified by the Wx algorithm were comparable to or outperformed the UGCBs identified by different methods. We further validated the performance by evaluating the classification accuracy of Wx-14-UGCB and Peng-14-UGCB with cancer and normal RNA-seq data from three independent cancer studies including a melanoma cohort (GSE72056) that had not been included in the 12 types of TCGA cancer cohort.

We calculated the classification accuracy by dividing the samples in a given cohort into the training set (2,888 samples, 64%), validation set (723 samples, 16%), and test set (902 samples, 20%). Then, the training set was used to train a model using a neural network (NN) algorithm and the validation set was used to assess how well the model had been trained. Finally, the test set was used to calculate the classification accuracy with the trained model.

The comparison revealed that Wx-14-UGCB classified malignant and non-malignant melanoma single cells better than Peng-14-UGCB. With the expression levels of the genes in the Wx-14-UGCB set, 818 out of the 902 test samples were correctly classified, whereas 633 out of 902 test samples were correctly classified using the Peng-14-UGCB set.

For the lung adenocarcinoma data set (GSE40419), Wx-14-UGCB showed 80.00% classification accuracy when classifying lung cancer and adjacent normal cells, while Peng-14-UGCB showed 56.87% classification accuracy.

For the 5,578 head and neck squamous cell carcinoma single cells (2,215 cancer cells and 3,363 non-cancer cells), the classification accuracy was 98.19% for Wx-14-UGCB and 87.20% for Peng-14-UGCB.

Table 3. Classification accuracy comparison (%).

<table>
<thead>
<tr>
<th>Type</th>
<th>Wx</th>
<th>Peng's edgeR</th>
<th>Martinez-Ledesma's</th>
</tr>
</thead>
<tbody>
<tr>
<td>BLCA</td>
<td>95.79</td>
<td>97.20</td>
<td>94.86</td>
</tr>
<tr>
<td>BRCA</td>
<td>98.19</td>
<td>96.38</td>
<td>91.78</td>
</tr>
<tr>
<td>COAD</td>
<td>94.51</td>
<td>87.20</td>
<td>98.78</td>
</tr>
<tr>
<td>HNSC</td>
<td>97.17</td>
<td>92.23</td>
<td>94.35</td>
</tr>
<tr>
<td>KICH</td>
<td>95.65</td>
<td>—</td>
<td>100.00</td>
</tr>
<tr>
<td>KIRC</td>
<td>99.67</td>
<td>—</td>
<td>99.34</td>
</tr>
<tr>
<td>KIRP</td>
<td>99.38</td>
<td>—</td>
<td>100.00</td>
</tr>
<tr>
<td>LIHC</td>
<td>90.57</td>
<td>94.81</td>
<td>87.74</td>
</tr>
<tr>
<td>LUAD</td>
<td>97.92</td>
<td>97.58</td>
<td>98.96</td>
</tr>
<tr>
<td>LUSC</td>
<td>98.19</td>
<td>96.75</td>
<td>99.28</td>
</tr>
<tr>
<td>PRAD</td>
<td>93.45</td>
<td>—</td>
<td>92.36</td>
</tr>
<tr>
<td>THCA</td>
<td>95.80</td>
<td>—</td>
<td>90.21</td>
</tr>
<tr>
<td>Total</td>
<td>96.72</td>
<td>94.59</td>
<td>94.81</td>
</tr>
</tbody>
</table>

Figure 2.5: Performance of the Wx-14-UGCB on BRCA, LUAD, and LUSC RNA-seq data. AUC values are listed. ROC, receiver operating characteristic.
was the best on 6 cancer types, while Peng-14-UGCB and edgeR outperformed on two and five sets, respectively. The comparison result of area under the curve (AUC) values for BRCA, LUAD, and LUSC were 0.9944, 0.9943, and 0.9936, respectively (Figure 2.5), showing excellent classification performance of the Wx-14-UGCB set.

![Figure 2.6](image6.png)

Figure 2.6: Classification accuracy according to given number of genes. The x-axis indicates the number of top genes (sorted in descending order by the DI values) used for the calculation and the y-axis represents the average accuracy.

![Figure 2.7](image7.png)

Figure 2.7: Comparison of genes identified by Wx and edgeR. The x-axis indicates the number of top genes used for the comparison and the y-axis represents the percentage of overlap between the gene sets.

As shown in Figure 2.6, approximately the top 100 UGCBs (Wx-100-UGCB) reached a plateau with the highest classification accuracy. We wondered how many genes identified by the Wx algorithm coincided with DEGs identified using edgeR.
Intriguingly, less than 35% of genes overlapped (Figure 2.7). For example, a comparison of the top 500 biomarker candidate genes identified by both algorithms showed that only 45 genes (9.0%) were common. In the case of top 2,000 genes, only 379 genes (19.0%) overlapped. Thus, there was substantial discrepancy between the algorithms with the same gene expression data. Next, we performed gene ontology (GO) and network analysis to investigate the putative function of top 50 UGCBs using Metascape [156]. Genes involved in the Fc gamma receptor dependent phagocytosis, antigen processing and presentation, and regulation of apoptotic signaling pathway were significantly altered (Figure 2.8), suggesting that the deregulation of these pathways might be a critical factor in the onset or progression of most cancers. Further investigations of these genes are warranted.

<table>
<thead>
<tr>
<th>GSE id</th>
<th>Cancer type</th>
<th>Wx-14-UGCB</th>
<th>Peng-14-UGCB</th>
</tr>
</thead>
<tbody>
<tr>
<td>GSE72056</td>
<td>Melanoma</td>
<td><strong>90.71</strong></td>
<td>70.22</td>
</tr>
<tr>
<td>GSE40419</td>
<td>Lung adenocarcinoma</td>
<td><strong>80.00</strong></td>
<td>56.87</td>
</tr>
<tr>
<td>GSE103322</td>
<td>Head and neck squamous cell carcinoma</td>
<td><strong>81.10</strong></td>
<td>68.28</td>
</tr>
</tbody>
</table>

Table 2.3: The classification accuracy of the UGCBs for non TCGA dataset (%).

We further validated the performance by evaluating the classification accuracy of Wx-14-UGCB and Peng-14-UGCB with cancer and normal RNA-seq data from three
independent cancer studies including a melanoma (GSE72056), Lung adenocarcinoma (GSE40419), and head and neck squamous cell carcinoma (GSE103322) that had not been included in the 12 types of TCGA cancer cohort [179, 154]. We calculated the classification accuracy by dividing the samples in a given cohort into the training set (2,888 samples, 64%), validation set (723 samples, 16%), and test set (902 samples, 20%). Then, the training set was used to train a model using a neural network (NN) algorithm and the validation set was used to assess how well the model had been trained. Finally, the test set was used to calculate the classification accuracy with the trained model. As shown in Table 2.3, Wx-14-UGCB significantly outperformed Peng-14-UGCB by large margin. Specifically, with the expression levels of the genes in the Wx-14-UGCB set, 818 out of the 902 test samples were correctly classified, whereas 633 out of 902 test samples were correctly classified using the Peng-14-UGCB set. For the lung adenocarcinoma data set (GSE40419), Wx-14-UGCB showed 80.00% classification accuracy when classifying lung cancer and adjacent normal cells, while Peng-14-UGCB showed 56.87% classification accuracy. For the 5,578 head and neck squamous cell carcinoma single cells (2,215 cancer cells and 3,363 non-cancer cells) (GSE103322), Wx-14-UGCB showed 81.10% classification accuracy when classifying cancer and non-cancer cells, while Peng-14-UGCB showed 68.28% classification accuracy. In summary, the top 14 genes (Wx-14-UGCB) identified by the Wx algorithm could potentially be used as novel gene expression biomarkers for the detection of various types of cancers, although its use might be limited by clinical difficulties associated with RNA-based applications. Further experimental investigations are required to validate the Wx-14-UGCB.
2.5 Discussion

The proposed algorithm estimates the classification power of genes in a given gene expression data set using the discriminative index (DI) score algorithm. Researchers can intuitively select gene-expression biomarker candidates from the DI scored gene list.

<table>
<thead>
<tr>
<th>Cancer type</th>
<th>Wx-14-UGCB</th>
<th>Peng-14-UGCB</th>
</tr>
</thead>
<tbody>
<tr>
<td>BLCA</td>
<td>EEF1A1, FN1, GAPDH, SFTPC, AHNK, KLK3, UMOD, CTSB, COL1A1, GPX3, GNAS, ATP1A1, SFTPB, ACTB</td>
<td>KIF4A, NUSAP1, HJURP, NEK2, FANCI, DTL, UHRF1, FEN1, IQGAP3, KIF20A, TRIM59, CENPL, C16orf59, UBE2C</td>
</tr>
<tr>
<td>BRCA</td>
<td></td>
<td></td>
</tr>
<tr>
<td>COAD(READ)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>HNSC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>KICH</td>
<td></td>
<td></td>
</tr>
<tr>
<td>KIRP</td>
<td></td>
<td></td>
</tr>
<tr>
<td>KIRC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>LIHC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>LUAD</td>
<td></td>
<td></td>
</tr>
<tr>
<td>LUSC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PRAD</td>
<td></td>
<td></td>
</tr>
<tr>
<td>THCA</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 2.4: Gene expression biomarkers identified by different studies.

Our interesting finding is that the 14 gene signatures (Wx-14-UGCB) identified by the Wx algorithm includes the housekeeping gene GAPDH (Table 2.4), which has been used in many studies as a control (or reference) gene. Recently, several concerns about using the GAPDH gene as a housekeeping gene has been reported [56, 46, 6, 146, 17]. Our result also indicated that the GAPDH gene was one of the highest DI-scored genes, and this gene should therefore be used with caution as a control gene in gene expression experiments. Interestingly, another well-known housekeeping gene ACTB was ranked 14 out of 20,501 genes (Table 2.5), suggesting that both GAPDH and ACTB genes might be unsuitable housekeeping genes for gene expression experiments, particularly in cancer studies. Further investigations of the remaining genes such as FN1, EEF1A1, COL1A1, SFTPB, SFTPC, and ATP1A1 will shed light on the identification of novel biomarker genes for a pan-cancer cohort.

One of the disadvantages of artificial neural network-based approaches when ap-
<table>
<thead>
<tr>
<th>Rank</th>
<th>Gene</th>
<th>Discriminative Index (arbitrary number, higher is better)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>EEF1A1</td>
<td>1.65849</td>
</tr>
<tr>
<td>2</td>
<td>FN1</td>
<td>1.61224</td>
</tr>
<tr>
<td>3</td>
<td>GAPDH*</td>
<td>1.50260</td>
</tr>
<tr>
<td>4</td>
<td>SFTPC</td>
<td>0.96081</td>
</tr>
<tr>
<td>5</td>
<td>AHNAK</td>
<td>0.71224</td>
</tr>
<tr>
<td>6</td>
<td>KLK3</td>
<td>0.56306</td>
</tr>
<tr>
<td>7</td>
<td>UMOD</td>
<td>0.55580</td>
</tr>
<tr>
<td>8</td>
<td>CTSB</td>
<td>0.42822</td>
</tr>
<tr>
<td>9</td>
<td>COL1A1</td>
<td>0.41349</td>
</tr>
<tr>
<td>10</td>
<td>GPX3</td>
<td>0.37308</td>
</tr>
<tr>
<td>11</td>
<td>GNAS</td>
<td>0.36476</td>
</tr>
<tr>
<td>12</td>
<td>ATP1A1</td>
<td>0.34630</td>
</tr>
<tr>
<td>13</td>
<td>SFTPB</td>
<td>0.33725</td>
</tr>
<tr>
<td>14</td>
<td>ACTB*</td>
<td>0.32997</td>
</tr>
<tr>
<td>15</td>
<td>ACPP</td>
<td>0.32805</td>
</tr>
<tr>
<td>16</td>
<td>FTL</td>
<td>0.31993</td>
</tr>
<tr>
<td>17</td>
<td>P4HB</td>
<td>0.31076</td>
</tr>
<tr>
<td>18</td>
<td>A2M</td>
<td>0.30867</td>
</tr>
<tr>
<td>19</td>
<td>PIGR</td>
<td>0.29527</td>
</tr>
<tr>
<td>20</td>
<td>DCN</td>
<td>0.29410</td>
</tr>
<tr>
<td>21</td>
<td>EEF2</td>
<td>0.28864</td>
</tr>
<tr>
<td>22</td>
<td>CLU</td>
<td>0.28477</td>
</tr>
<tr>
<td>23</td>
<td>ACTG1</td>
<td>0.25872</td>
</tr>
<tr>
<td>24</td>
<td>PABPC1</td>
<td>0.24866</td>
</tr>
<tr>
<td>25</td>
<td>SPARC</td>
<td>0.24861</td>
</tr>
<tr>
<td>26</td>
<td>CTSB</td>
<td>0.24328</td>
</tr>
<tr>
<td>27</td>
<td>RPL3</td>
<td>0.23105</td>
</tr>
<tr>
<td>28</td>
<td>RPL8</td>
<td>0.22458</td>
</tr>
<tr>
<td>29</td>
<td>ALDOA</td>
<td>0.21630</td>
</tr>
<tr>
<td>30</td>
<td>B2M</td>
<td>0.21391</td>
</tr>
<tr>
<td>31</td>
<td>MYH11</td>
<td>0.21365</td>
</tr>
<tr>
<td>32</td>
<td>TPT1</td>
<td>0.20991</td>
</tr>
<tr>
<td>33</td>
<td>HLA-B</td>
<td>0.20859</td>
</tr>
<tr>
<td>34</td>
<td>TXNIP</td>
<td>0.20725</td>
</tr>
<tr>
<td>35</td>
<td>HSP90AB1</td>
<td>0.20676</td>
</tr>
<tr>
<td>36</td>
<td>MGP</td>
<td>0.20396</td>
</tr>
<tr>
<td>37</td>
<td>APP</td>
<td>0.20064</td>
</tr>
<tr>
<td>38</td>
<td>PKM2</td>
<td>0.19627</td>
</tr>
<tr>
<td>39</td>
<td>ALB</td>
<td>0.19292</td>
</tr>
<tr>
<td>40</td>
<td>ALDOB</td>
<td>0.19162</td>
</tr>
<tr>
<td>41</td>
<td>KRT13</td>
<td>0.18497</td>
</tr>
<tr>
<td>42</td>
<td>C4A</td>
<td>0.18036</td>
</tr>
<tr>
<td>43</td>
<td>CALR</td>
<td>0.17827</td>
</tr>
<tr>
<td>44</td>
<td>APLP2</td>
<td>0.17746</td>
</tr>
<tr>
<td>45</td>
<td>ENO1</td>
<td>0.17689</td>
</tr>
<tr>
<td>46</td>
<td>HLA-A</td>
<td>0.17441</td>
</tr>
<tr>
<td>47</td>
<td>GSN</td>
<td>0.17034</td>
</tr>
<tr>
<td>48</td>
<td>COL1A2</td>
<td>0.16909</td>
</tr>
<tr>
<td>49</td>
<td>MYH9</td>
<td>0.16713</td>
</tr>
<tr>
<td>50</td>
<td>APOE</td>
<td>0.16215</td>
</tr>
</tbody>
</table>

Table 2.5: Genes ranked by the discriminative index score. * indicates genes known as house keeping genes.
Table 2.6: Classification accuracy (%) of non-cancer transcriptomic data set.

| Number of top features (genes) identified by WX | Classifier | Liver (GSE105127) |  
|  |  | Pericentral (n=19) | Intermediate (n=19) | Periportal (n=19) |
|---|---|---|---|
| 10 | XGBoost | 57.89 | 50.00 | 50.00 |
|  | SVM | 68.42 | 72.22 | 0.00 |
| 20 | XGBoost | 78.95 | 66.67 | 50.00 |
|  | SVM | 73.68 | 44.44 | 0.00 |
| 30 | XGBoost | 84.21 | 72.22 | 62.50 |
|  | SVM | 73.68 | 27.78 | 6.25 |
| 50 | XGBoost | 84.21 | 66.67 | 50.00 |
|  | SVM | 78.95 | 33.33 | 0.00 |
| 100 | XGBoost | 84.21 | 61.11 | 62.50 |
|  | SVM | 84.21 | 55.56 | 0.00 |
| 200 | XGBoost | 84.21 | 77.78 | 81.25 |
|  | SVM | 84.21 | 61.11 | 0.00 |

Applied to biomedical data is that a large number of samples are needed to achieve good classification or regression performance. We observed relatively lower classification accuracy (Table 2.6) when the Wx algorithm was applied to a non-cancer transcriptomic data set (GSE105127), which contains normalized expression levels of 65,671 transcripts performed in the pericentral ($n = 19$), intermediate ($n = 19$), and periportal ($n = 19$) regions of the human liver isolated by laser-captured microdissection [14]. In addition, selected features from the same data set vary depending on algorithms. In our comparison, there were no overlaps between the top 14 genes identified by Wx or Pengs. This kind of inconsistency is caused mainly by the algorithmic difference, as reported in several differentially expressed gene analysis studies [50, 29, 164]. Thus, it is difficult to establish which algorithm is better by comparison without experimental verification. Therefore, the usefulness of the 14 genes (Wx-14-UGCB) for cancer biomarkers should be validated with extensive experimental evidence in the near future.

In summary, the Wx algorithm developed in this study estimates the classification
power of genes in a given gene expression data set using the discriminative index (DI) score algorithm. Researchers can intuitively select gene-expression biomarker candidates from the DI scored gene list. Further experimental validation will be necessary to prove the Wx algorithms usefulness.

2.6 Contribution

In this project, my contributions are as follows.

- Designed the study with Park and Kang.
- Developed the algorithm with Park and Shim.
- Analyzed the result with Ki.Kang, Park, Ke.Kang, and Choi.
- Wrote the manuscript with Ki.Kang, Park, Ke.Kang, Shim, and Choi.
3

Prognosis-Related Target Identification

In the second biomarker identification project, we aim to extend the first feature selection method (Chapter 2) to a regression task. The first feature selection method (Chapter 2) is dealing with a discrete target label, while the target value of this project is a survival time, which is continuous. Therefore, we cannot directly use the previous method in a prognosis-related feature selection problem. To solve this problem, we present a new feature selection method called Cascaded Wx algorithm.

3.1 Motivation

In this project, the goal is to select a concise subset of most important genes from the whole 20,000 genes, the result of RNA sequencing data that can classifying high/low risk cohorts. We focus on lung cancer patients because lung cancer is the most commonly diagnosed cancer and the second most common cause of cancer-related deaths worldwide [12]. Most lung cancer cases are nonsmall cell lung cancer (NSCLC), and lung adenocarcinoma (LUAD) accounts for more than 50% of all NSCLCs. Recently, survival rates for LUAD patients have been greatly improved with the development
of improved treatment approaches, including surgical or radiation techniques, and the introduction of targeted therapies and immunotherapies tailored to the molecular or immunologic characteristics of tumors. However, the survival rate is still only about 50% for potentially curatively resected LUAD [171]. To optimize clinical intervention, it is important to identify which patients have poor prognoses. The prediction of prognosis requires an extensive knowledge of various aspects of cancer biology and an understanding of relevant clinical information such as TNM stage, histology, and genetic mutations [61]. Among the clinical features, TNM staging is the most successful clinical parameter in practice and is widely used to predict patients prognoses. However, this staging method still has room for improvement in the era of genomic sequencing, where abnormalities in multiple genes can be detected simultaneously [134]. Among the various genome-wide applications, the gene expression signature is the most promising approach to the prediction of clinical outcomes [159, 128, 24], as a suite of expressed genes reflects the identity of a given cell population. Several gene expression-based clinical applications such as MammaPrint [168] and Oncotype DX [18] are being used in clinical practice. These applications predict patients’ prognoses and drug and/or chemotherapy responsiveness by examining the expression levels of a defined gene set. Therefore, the identification of a particular gene set associated with clinical findings is crucial in many disease research studies.

Recent technological advancements in clinical genome sequencing using next-generation sequencing (NGS) technologies provide opportunities to understand the relationships between gene expression and tumor phenotypes [86]. For example, several studies classify NSCLC patients into subgroups with differing clinical outcomes using gene signatures [22, 148, 11, 173]. However, the results of such studies have been unsatisfactory in terms of discrepancies between identified gene signatures. The possible reasons for the inconsistent results among the studies include the use of small samples compared to the number of genes (high-dimensional data), the use of differ-
ent platforms, and the problems with feature preprocessing steps. In addition, there are no robust methods for analyzing such high-dimensional data effectively.

Machine learning (ML) algorithms can be a useful approach to the analysis of high volumes of data if a model is well constructed with high-quality input data for training. Numerous variations of the original ML algorithms have been developed and applied to a variety of problems [95, 118, 183, 49]. In molecular biology, NGS technologies, which revolutionized the profiling approach by sequencing huge numbers of given short DNA fragments, have been generating enormous amounts of data these days [60]. Because of this, there is an urgent need to develop ML-based algorithms that can effectively analyze such high volumes of genomic data. Support vector machines (SVM; [21]), k-nearest neighbors [30], multilayer perceptrons [103], decision trees [25], random forest (RF; [182]) algorithms, logistic regression, and gradient boosting machines [102] are ML algorithms that are frequently used to analyze big data. However, these methods were not originally designed to extract prognostic features from patients’ data. Recently, several ML-based algorithms have been proposed to select a subset of key features (genes) for classification [3, 176, 52] or to identify prognostic features [167] from high-throughput molecular profiling data. There is still room for improvement, however, as new deep learning algorithms continue to emerge in the field of ML [41, 122].

To effectively analyze multidimensional datasets, dimension-reduction algorithms such as feature selection are often required. Principal component analysis (PCA; [169]), nonnegative matrix factorization [90], kernel PCA [107], graph-based kernel PCA, linear discriminant analysis [106], and generalized discriminant analysis [7] are algorithms that are widely applied to high-dimensional biomedical datasets. In addition to these approaches, several studies recently used artificial neural networks to predict clinical outcomes in lung cancer patients [77, 172, 64]. However, these approaches do not fully take into account available information such as high-throughput
profiling data (e.g., transcriptomes) and/or clinical information for feature selection. To address these problems, we developed a novel feature selection framework called Cascaded Wx (CWx) to enhance the efficiency of feature selection and the accuracy of prediction for given patients prognosis. Our analyses revealed that the CWx framework selected more prognosis-related features than algorithms in categories such as similarity-based, sparse learning-based, ML-based, and statistical-based models, highlighting the potential value of our proposed framework for biomedical data.

3.2 Problem Definition

Prognosis-related Target Identification Task

- **Data**: Gene data with survival time
  - $X$: 20,000 different genes (features)
  - $Y$: Time to event (decease)

- **Task**: Select top $k$ features that contributes most in classifying high/low risk cohorts.

Gene expression data (mRNASeq) of The Cancer Genome Atlas (TCGA) lung adenocarcinoma (LUAD) were obtained from Firehose at The Broad Institute (https://gdac.broadinstitute.org). From the whole data, we extracted gene features, survival values, and censoring information, which can be formally represented as $X \in \mathbb{R}^{n \times d}$, $S \in \mathbb{R}^n$, and $C \in \mathbb{R}^n$, respectively; $n$ is the number of patients, and $d$ is the feature dimension. If $C_i = 0$ (uncensored patients), the survival time interval $(S_i)$ represents the time between the start of observing the patient status and the
event time (date of death). If a patient datum is right censored ($C_i = 1$), the survival time interval ($S_i$) represents the time elapsed between the start of observing the patient status and the end of the study. Among the 507 LUAD patients, there are 183 uncensored (death event occurred) samples and 324 right-censored samples. Each sample contained read counts (expression levels) of 20,501 genes. These count-based values are abundant for a few specific transcripts (highly expressed genes), a factor that prevents a model from finding a good pattern. To mitigate this problem, we use a log transformation:

$$X_{ij}^{new} = \log_2(X_{ij} + 1)$$

for $i \in n$ and $j \in d$. A constant, 1, is added to the read count value of each gene before applying the logarithm function to avoid the numerical problems. Min-max normalization is then applied to the log-transformed data. With these datasets, the objective is to select the optimal gene set associated with patients' prognoses using the survival information.

### 3.3 Proposed Model: Cascaded Wx

The proposed method was based on the Wx algorithm [118] (Chapter 2), which identifies key genes discriminating different classes. As this method was designed for a classification problem, we extend it to be applicable to the patient cohort grouping task with continuous survival values. The basic concept of the proposed algorithm (cascaded WX, CWx) is to guide the feature selection algorithm by efficiently organizing the training curriculum from an easy to hard one. For each stage we select a subset of the training samples (patients). There are multiple stages with different difficulty levels. By doing this, the model will automatically reduce the number of features (genes). The three stage example is summarized in (Figure 3.1). In the first step, patients are divided into high- and low-risk cohorts according to whether
Figure 3.1: An example of CWx’s feature selection procedure. Input samples (patients) are reduced through three cascaded steps using different criteria. For each step, a different cutoff criteria is used in an easy to hard manner. For example, three-year, 2 versus 4-year, and 1 versus 5-year cutoffs for grouping samples into either high- or low-risk groups are used at the first, second, and third steps, respectively. Input features (genes) are also reduced by a quarter in each step. Finally, the prognostic potential of features can be estimated according to the weights calculated from the trained neural network.

they have survived for 3 years. For example, 115 deceased patients within 3 years in a training set formed one group (28.4%; high risk), whereas 104 patients who lived more than 3 years formed another group (25.7%; low risk). The remaining patients (186, 45.9%) were right censored, meaning that there was no information as to whether these patients were deceased within 3 years. These right-censored patients are excluded in the training stage. The second and third steps are similar to the first step with different cutoffs (2 versus 4 years and 1 versus 5 years, respectively). As the number of samples is decreased by each step’s criteria, the number of features (genes) was also reduced by a quarter in each step. One quarter of the features were selected according to the importance determined by our previous Wx feature selection algorithm. A total of 19,960 genes were used as input features after removing genes with no variance. The final output after these three steps is a set of genes ranked by prognostic weights, estimated in a manner similar to the Wx algorithm.


3.4 Experiments

We compared the proposed algorithm, CWx, to the following supervised feature selection algorithms from five different categories: i) ML based models: RF [13], SVM [28], and Extreme Gradient Boosting (XGBoost) [23]; ii) similarity based models: fisher score [45], ReliefF [87], and trace ratio (Trace ratio) [110]; iii) sparse learning-based models: multi-task feature learning via efficient l2,1-norm minimization (LLL21) [96] and robust feature selection (RFS) [111]; iv) statistical based models: Fscore; and V) others : cox proportional hazard (CoxPH) [31]. These algorithms calculate a score for each given feature, so the performance of each cancer prognosis prediction can be estimated by comparing the highest-scoring features selected by each algorithm.

We also compared CWx to CoxPH and Coxnet as baseline methods for prognosis prediction. Feature selection criteria for CoxPH and Coxnet were P value and beta coefficients, respectively.

Figure 3.2: Violin plot of the comparison of feature selection algorithms with top 100 genes. The metric is a c-index with the selected 100 genes in lung adenocarcinoma (LUAD) samples for each algorithm (left). White circles indicate the medians; box limits inside the polygons indicate the 25th and 75th percentiles as determined by R software; whiskers extend 1.5 times the interquartile range from the 25th and 75th percentiles; polygons represent density estimates of data and extend to extreme values. Asterisks (*p < 0.05 and ***p < 0.001) indicate the results of one-way ANOVA (p < 0.0001) with post hoc test (pairwise t test with BonferroniHolm correction). x- and y-axes indicate the number of cumulative top genes and c-index, respectively (right).

The results indicated that CWx was superior to the other methods in terms of
c-index when comparing the top genes (cumulative) from 1 to 100 in LUAD samples (Figure 3.2).

The CWx framework was designed to select the optimal gene set associated with patients' prognoses using the survival information of a given cohort and changing the separation criteria between high- and low-risk groups through a three-step cascade method. Therefore, the CWx algorithm can be applied to the identification of prognosis-related genes associated with a range of diseases, not only LUAD (Figure S1). In addition, CWx has a linear execution time to complete the feature selection steps depending on the number of samples. Some information theoretical-based feature selection algorithms take longer to finish the feature selection procedure. In contrast, one of the disadvantages of CWx is that it can only handle right-censored data within 3 years due to the binary classification of patients into either high- or low-risk groups. However, all of the supervised feature selection algorithms have this problem when applied to survival analysis. One possible solution to this issue is to select features directly from a given neural network training model using a negative log-likelihood cost function that can handle the whole sample for survival analysis.

One of the key pathways related to the prognosis of LUAD patients identified by the CWx framework was the Wnt signaling pathway. A recent study has shown that two distinct subpopulations of cells, one with high Wnt signaling activity and another forming a niche that provides the Wnt ligand, are activated in LUAD. In addition, in vitro and in vivo studies have suggested that Wnt responsiveness contributes to the survival of cancer cells and the maintenance of a stem cell-like niche cell phenotype (Tammela et al., 2017). Interestingly, several prognosis-related genes identified by the CWx framework have been previously reported in LUAD studies. For example, glycogen synthase kinase 3 is a central regulator of cellular metabolism, development, and growth and is frequently

Next, we performed Gene Ontology (GO) analysis to identify the biological pathways associated with the top 100 genes. GO analysis was performed using Metascape (http://metascape.org/gp/index.html) with top 100 genes (default parameters were used). The significance of a given GO term is represented by gray (significant) or white (nonsignificant) bars with a P cutoff value of 0.0001.

Figure 3.3: Gene ontology (GO) analysis of top 100 genes. GO analysis was performed using Metascape (http://metascape.org/gp/index.html) with top 100 genes (default parameters were used). The significance of a given GO term is represented by gray (significant) or white (nonsignificant) bars with a P cutoff value of 0.0001.
the key pathways regulating development, and closely associated with many cancers. The gene sets identified by the other algorithms were related to different pathways such as “positive regulation of JNK cascade” (CoxPH), “central carbon metabolism in cancer” (Fisher score and Fscore), “O-glycan biosynthesis, mucin type core” (LLL21, RF, and XGBoost), “mitotic nuclear division” (Trace ratio), “regulation of gene silencing” (RFS), and “GPCR ligand binding” (SVM). Differences between the gene sets identified by the different algorithms, and their associated biological pathways, need to be further investigated in future studies.

3.5 Discussion

Our evaluation using 507 TCGA LUAD transcriptomes revealed that the proposed model outperformed the other methods by effectively training the algorithm from an easy to hard problem. This finding means that the gene set found by CWx is one of the best candidate gene sets to predict patients’ prognoses. In addition, CWx has a linear execution time to complete the feature selection steps depending on the number of samples. While the information theoretical-based feature selection algorithms take longer to finish the feature selection procedure.

One of the key pathways related to the prognosis of LUAD patients identified by the CWx framework was the Wnt signaling pathway. A recent study has shown that two distinct sub-populations of cells, one with high Wnt signaling activity and another forming a niche that provides the Wnt ligand, are activated in LUAD. In addition, in vitro and in vivo studies have suggested that Wnt responsiveness contributes to the survival of cancer cells and the maintenance of a stem cell-like niche cell phenotype [151]. Not only that, many CWx found genes have been previously reported in LUAD studies [161, 101, 78, 1, 139, 51, 177].
3.6 Contribution

In this project, my contributions are as follows.

- Designed the study with Park, Ko, and Kang.
- Developed the algorithm with Park.
- Wrote the manuscript with Park, Hong, Ko, and Kang.
4

Drug Repurposing

The molecule discovery is divided into two tasks: drug repurposing (Chapter 4) and optimized drug candidate generation (ODG, Chapter 5). The result of the drug repurposing project have been published to the machine learning for healthcare conference [144]. In addition, one of the case studies has been submitted to Computational and Structural Biotechnology Journal.

4.1 Introduction

Many diseases are caused by abnormal protein levels, therefore, a drug is designed to target particular proteins. However, a drug may not work well for a decent portion of patients, because an individual’s response to a drug varies depending on the genetic inheritance [163]. Unfortunately, pharmaceutical companies focus only on a majority cohort of patients as drug discovery is an expensive process. The reduction of the cost of the drug discovery process will not only lead to drugs costing less, resulting in reduced healthcare costs for a patient but can also allow companies to develop personalized drugs based on genetics.

Among the many parts of the drug discovery process, predicting drug-target interactions (DTI) is an essential one. DTI is difficult and costly as experimental assays
not only take significant time but are expensive. Furthermore, only less than 10% of the proposed DTIs are accepted as new drugs [66]. Therefore, in silico (performed on a computer) DTI predictions are much demanded since it can expedite the drug development process by systemically suggesting a new set of candidate molecules promptly, which can save time and reduce the cost of the whole process by up to 43% [43].

In response to this demand, three types of in silico DTI prediction methods have been proposed in the literature: molecular docking, similarity-based, and deep learning-based models. Molecular docking [157, 100] is a simulation-based method using the 3D structured features of molecules and proteins. Although it can provide an intuitive visual interpretation, it is difficult to obtain a 3D structure of a feature and cannot scale to large datasets. To mitigate these problems, two similarity-based methods, KronRLS [116] and SimBoost [66] have been proposed using efficient machine learning methods. However, using a similarity matrix has two downsides. Firstly, feature representation is limited in the similarity space, thereby ignoring the rich information embedded in the molecule sequence. For example, if a brand new molecule is tested, the model will represent it using relatively unrelated (dissimilar) molecules, which would make the prediction inaccurate. Secondly, it necessitates the calculation of the similarity matrix which can limit the maximum number of molecules in the training process. To overcome these limitations, a deep learning-based DTI model, DeepDTA [114], was proposed. It is an end-to-end convolutional neural network (CNN)-based model that eliminates the need for feature engineering. The model automatically finds useful features from a raw molecule and protein sequence. Its success has been demonstrated on two publicly available DTI benchmarks. Although this work illustrated the potential of a deep learning-based model, there are several areas for improvement:

- CNNs can’t model potential relationships among distant atoms in a raw molecule
sequence. For example, with three layers of CNNs each with a filter size of 12, the model can capture associations in atoms up to 35 distances in a sequence. We posit that the recently proposed self-attention mechanism can be used to capture any relationship among atoms in a sequence, and thereby provide a better molecule relationship

- The one-hot encoding used to represent each molecule fails to take advantage of existing chemical structure knowledge. An abundance of chemical compounds are available in the PubChem database [55], from which we can extract useful chemical structures for pre-training the molecule representation network.

- Fine-tuning is a type of transfer learning where weights trained from one network can be transferred to another so that the weights can be adjusted to the new dataset. Thus, we can transfer the weights learned from the PubChem database to our DTI model. This will help our model to use the learned knowledge of a chemical structure while tailoring it to predicting DTI interactions.

With these observations, we propose a new deep DTI model, Molecule Transformer DTI (MT-DTI), based on a new molecule representation. We use a self-attention mechanism to learn the high-dimensional structure of a molecule from a given raw sequence. Our self-attention mechanism, Molecular Transformer (MT), is pre-trained on publicly available chemical compounds (PubChem database) to learn the complex structure of a molecule. This pre-training is important, because most datasets available for DTI training has only 2000 molecules, while the data for pre-training (PubChem database) contains 97 millions of molecules. Although it does not contain interaction data but just molecules, our MT is able to learn a chemical structure from it, which will be effectively utilized when transferred to MT-DTI (our model). Therefore, we transfer this trained molecule representation to our DTI model so that it can be fine-tuned with a DTI dataset. The proposed DTI model is evaluated on two
well-known benchmark DTI datasets, Kiba [152] and Davis [35], and outperforms the current state of the art (SOTA) model by 4.9% points for Kiba and 1.6% points for Davis in terms of area under the precision-recall curve. Additionally, we demonstrate the usefulness of our trained model using a known drug list targeting a specific protein. The trained model generates all FDA approved drugs with high rankings in the drug candidate lists. The demonstrated effectiveness of the proposed model can help reduce the cost of drug discovery. Furthermore, precise molecule representation can enable drugs to be designed for specific genotypes and potentially enable personalized medicine.

**Technical Significance**  We propose a novel molecule representation, adapting the self-attention mechanism that was recently proposed in Natural Language Process (NLP) literature. This is inspired by the idea that understanding a molecule sequence for a chemist is analogous to understanding a language for a person. We introduce a new way to train the molecule representation model to fit the DTI problem using an existing corpus to achieve a more robust representation. With this (pre)trained molecule representation, we fine-tune the proposed DTI model and achieve new SOTA performances on two public DTI benchmarks.¹

**Clinical Relevance**  With our new model, we can potentially lower medication costs for patients, which can help make drugs more affordable and help patients be more adherent. In addition, this can serve as the stepping stone for designing personalized medication. Through the proper representation of molecules and proteins, we can better understand the properties of patients that make a drug helpful or not [127].

¹The demo is publicly available at: https://mt-dti.deargendev.me/
4.2 Related Work

Predicting drug-target interaction traditionally focused on a binary classification problem [174, 10, 158, 15, 58, 27, 16, 113]. The most recent approach tackling this binary classification problem is an interpretable deep learning based model [53]. Although these methods show promising results on binary datasets, they are simplifying protein-ligand interactions by thresholding affinity values. In order to model these complex interactions, several methods have been proposed, which can be categorized into three kinds. The first category of these models is molecular docking [157, 100], which is a simulation-based method. These methods are not scalable, due to heavy preprocessing. To overcome this downside, the second category, similarity-based methods, was proposed. They are KronRLS [116] and SimBoost [66], which is based on the calculation of similarity matrix of inputs. With the advent of deep learning, two deep learning-based methods have been proposed [53, 114]. Like these models, our model is also based on deep learning, but our proposed model has a better molecule representation, and improves its performance through a transfer learning technique.

Deep learning-based transfer learning, pre-training and fine-tuning, have been applied to various tasks such as computer vision [133, 54], NLP [71], speech recognition [75, 99], and health-care applications [141]. The idea is to use appropriate pre-trained weights to improve results in corresponding tasks, which also can be found in our experimental results.
4.3 Problem Definition

Drug Target Interaction

- **Data**: Molecules in SMILES format and proteins in FASTA format
  - **X**: molecule and protein
  - **Y**: affinity score

- **Task**: Predict a affinity score for a given pair of inputs.

4.4 Proposed Model: Molecule Transformer-Drug Target Interaction

We introduce a new drug-target interaction (DTI) model and a new molecule representation in this section. The basic motivation of the proposed model is that the structure of molecule sequences is shown to be very similar to the structure of natural language sentences in that contextual and structural information of atoms are important when understanding the characteristics of a molecule [76]. Specifically, each atom interacts with not only neighboring atoms but also long distant ones in a simplified molecular-input line-entry system (SMILES) sequence, a notation that encodes the molecular structure of chemicals. However, the current SOTA method using CNNs can’t relate long distance atoms when representing a molecule. We overcome this using the self-attention mechanism. We first describe the proposed MT-DTI model architecture (Figure 4.1) with input and output representation. We then elaborate on each of the three main building blocks of our MT-DTI model, the character-embedded Transformer layers (Molecule Transformers, Figure 4.2a, Section 4.4.2), the character-
Figure 4.1: The Proposed DTI Model Architecture. Inputs are molecule (SMILES) and protein (FASTA) and the regression output is the affinity score between these two inputs.

embedded Protein CNN layers (Protein CNNs, Figure 4.2b, Section 4.4.3), and the dense layers to model interactions between a drug and a protein (Interaction Denses, Figure 4.2c, Section 4.4.4). Then, we explain the process for pre-training the molecule transformers (MT) (Section 4.4.2).

### 4.4.1 Model Architecture

The MT-DTI model takes two inputs: a molecule represented by the SMILES [165] sequence and a protein represented by the FASTA [94] sequence. A molecule represented using the SMILES sequence is comprised of characters representing atoms or structure indicators. Mathematically, a molecule is represented as $I_M = \{m_1, m_2, \ldots, m_{L_M}\}$, where $m_i$ could be either an atom or a structure indicator, and $L_M$ is the sequence length, which varies depending on a molecule. This molecule sequence is fed into the Molecule Transformers (Section 4.4.2), to produce a molecule encoding, $M_{enc} \in \mathbb{R}^{E_M}$. Another type of input, a protein with FASTA sequence, also consists of characters of various amino acids. A formal protein representation is $I_P = \{p_1, p_2, \ldots, p_{L_P}\}$, where $p_j$ is one of the amino acids, and $L_P$ is the sequence length, which varies depending on a protein. This protein sequence is the input of the Protein CNNs (Section 4.4.3) and
generates a protein encoding, $P_{\text{enc}} \in \mathbb{R}^{E_P}$. Note that the encoding vector dimension $E_M$ and $E_P$ are model parameters. Both of the encodings, $M_{\text{enc}}$ and $P_{\text{enc}}$ are together fed into the multi-layered feed-forward network, Interaction Denses (Section 4.4.4), followed by the last regression layer, which predicts the binding affinity scores.

![Diagram of the proposed model](image)

(a) Molecule Trans. (b) Protein CNNs. (c) Interaction Denses.

Figure 4.2: Three parts of the proposed model.

### 4.4.2 Molecule Transformers

Molecule Transformers (Figure 4.2a) are multi-layered bidirectional Transformer encoders based on the original Transformer model [160]. The Transformer can model a sequence by itself without using a recurrent neural network (RNN) or CNN. Unlike these previous sequence processing layers (RNN or CNN), Transformer can effectively
encode the relationship among long-distance tokens (atoms) in a sequence. This powerful context modeling enables many Transformer-based NLP models to outperform previous methods in many benchmarks [160, 41]. Molecule Transformers is a modification of the existing Transformer, BERT [41], to better represent a molecule by changing the cost function. Before plugging it into the proposed model (Figure 4.1), we pre-train it using the modified masked language model task, which was introduced in the BERT model [41]. Each Transformer block consists of a self-attention layer and feedforward layer, and it takes embedding vectors as an input. Therefore the first Transformer block needs to convert an input sequence into the form of vectors using the input embedding.

**Input Embedding**

The input to the Molecule Transformers is the sum of the token embeddings and the position embeddings. The token embeddings are similar to the word embeddings [108], in that each token, $m_i$ is represented by a molecule token embedding (MTE) vector, $e_i$. These vectors are stored in a trainable weights MTE ∈ $\mathbb{R}^{V_M \times D_M}$, where $V_M$ is the size of the SMILES vocabulary and $D_M$ is the molecule embedding size. A MTE vector itself is not sufficient to represent a molecule sequence with a self-attention network, because a self-attention doesn’t consider the sequence order when calculating the attentions, unlike other attention mechanisms. Therefore, we add a trainable positional embedding (PE)², $p_i \in \mathbb{R}^{L_{max} \times D_M}$, to $e_i$ that makes the final input representation, $x_i$ where $L_{max}$ is the maximum length of a molecule sequence, which is set to 100 in this study. This process is illustrated in Figure 4.3.

We add five special tokens to the SMILES vocabulary to make a raw molecule sequence compatible with our model. [PAD] is for dummy padding to ensure the sequence has a fixed length. [REP] is a representation token that is used when fine-

²Please refer to [41] for more details.
<table>
<thead>
<tr>
<th>Token</th>
<th>[REP]</th>
<th>[BEGIN]</th>
<th>C</th>
<th>N</th>
<th>=</th>
<th>C</th>
<th>=</th>
<th>O</th>
<th>[END]</th>
</tr>
</thead>
<tbody>
<tr>
<td>MTE</td>
<td></td>
<td></td>
<td>e₁</td>
<td>e₂</td>
<td>e₃</td>
<td>e₄</td>
<td>e₅</td>
<td>e₆</td>
<td>e₇</td>
</tr>
<tr>
<td>PE</td>
<td>p₁ +</td>
<td>p₂ +</td>
<td>p₃ +</td>
<td>p₄ +</td>
<td>p₅ +</td>
<td>p₆ +</td>
<td>p₇ +</td>
<td>p₈ +</td>
<td>p₉ +</td>
</tr>
<tr>
<td>Model Input</td>
<td>x₁</td>
<td>x₂</td>
<td>x₃</td>
<td>x₄</td>
<td>x₅</td>
<td>x₆</td>
<td>x₇</td>
<td>x₈</td>
<td>x₉</td>
</tr>
</tbody>
</table>

Figure 4.3: An example of molecule token embedding (MTE) and positional embedding (PE) to make the model input $x_i$ for a given molecule sequence of methyl isocyanate (CN=C=O).

tuning the Transformer in the proposed MT-DTI model. [BEGIN]/[END] indicate the beginning or end of the sequence. These tokens are useful for the model when dealing with a sequence longer than $L_M^{\text{max}}$. When it is truncated on both sides, the absence of [BEGIN]/[END] tokens serve as an effective indicator of a truncation. Methyl isocyanate (CN=C=O), for example, can be represented with 9 tokens;

$$[\text{REP}] \ [\text{BEGIN}] \ C \ N = C = O \ [\text{END}]$$

Each token is transformed into a corresponding vector by referencing MTE and PE.

**Self-Attention Layer**

These transformed input vectors, $x_i$, are now compatible with an input to a self-attention layer. Each self-attention layer is controlled by a query vector ($q_i$), key vector ($k_i$), and value vector ($v_i$), where $i \in \{0, 1, \ldots, L_M^{\text{max}}\}$, all of which are different projections of the input, $X$ ($x_i \in \mathbb{R}^{L_M^{\text{max}} \times D_M}$), using trainable weights, $W_Q \in \mathbb{R}^{D_M \times D_Q}$, $W_K \in \mathbb{R}^{D_M \times D_K}$, and $W_V \in \mathbb{R}^{D_M \times D_V}$, shown correspondingly in Figure 4.4a. Then, the attention weights are computed as:
\[
Z = \text{Attention}(Q, K, V) \\
= \text{softmax} \left( \frac{QK^T}{\sqrt{D_k}} \right) V \in \mathbb{R}^{L_{\text{max}} \times D_v}
\]

\(D_k\) is the dimension of the key (one of the \(Z\)’s in Figure 4.4b). Thus, the learned relationship between the atoms can span the entire sequence via the self-attention weights.

**Feed-Forward Layer**

Similar to multiple filters in convolutional networks, a Transformer can have multiple attention weights, called multi-head attention. If one model has \(H\)-head attention, then it will have \(Z_h = \text{Attention}(XW_h^Q, XW_h^K, XW_h^V)\), where \(h \in \{1, 2, \ldots, H\}\). These \(H\) number of attention matrices, \(Z_h\), are then concatenated (shown on the left of Figure 4.4c) and projected using \(W^O \in \mathbb{R}^{H \cdot D_v \times D_M}\) (shown on the middle of Figure 4.4c) to form a final output of a Transformer, \(X^\text{out} \in \mathbb{R}^{L_{\text{max}} \times D_M}\) (shown on the right of Figure 4.4c).
pre-training

We adopt one of the pre-training tasks of BERT [41], the Masked Language Model. Since the structure of molecule sequences are shown to be very similar to the structure of natural language sentences [76] and there are abundant training examples, we hypothesize that predicting masked tokens is an effective way of learning a chemical structure. We adopt a special token, [MASK], for this task. It replaces a small portion of tokens so that the task of the pre-training model is to predict the original tokens. We choose 15% of SMILES tokens at random for each molecule sequence, and replace the chosen token with one of the special tokens, [MASK] with the probability of 0.8. For the other 20% of the time, we replace the chosen token with a random SMILES token\(^3\) or preserve the chosen token, with an equal probability, respectively. The target label of the task is the chosen token with the index. For example, one possible prediction task for Methyl isocyanate (CN=C=O) is

\[
\text{input : [REP] [BEGIN] C N = [MASK] = O [END]}
\]

\[
\text{label : (C, 5)}
\]

Fine-tuning

The weights of the pre-trained Transformers (Section 4.4.2) are used to initialize the Molecule Transformers in the proposed MT-DTI model (Figure 4.1). The output of the Transformers is a set of vectors, where the size is equivalent to the number of tokens. To obtain a molecule representation with a fixed length vector, we utilize the vector of the special token, [REP] in the final layer. This vector conveys the comprehensive bidirectional encoding information for a given molecule sequence, denoted as \(M^{rep} \in \mathbb{R}^{D_M}\).

\(^3\)Since the [MASK] token does not exist when testing, we need to occasionally feed irrelevant tokens when training.
4.4.3 Protein CNNs

Another type of input to the proposed MT-DTI model is a protein sequence. We modified the protein feature extraction module introduced by [114] by adding an embedding layer for the input.\(^4\) It consists of multi-layer CNNs with an embedding layer to make a sparse protein sequence continuous, and a pooling layer to represent a protein as a fixed size vector. For a given protein sequence, \(I_p\), each protein token, \(p_j\) is converted to a protein embedding vector by referencing trainable weights, \(PTE \in \mathbb{R}^{V_P \times D_P}\), where \(V_P\) is the size of the FASTA vocabulary and \(D_P\) is the protein embedding size. Let \(P \in \mathbb{R}^{L_{P}^{\text{max}} \times D_P}\) be a matrix representing the input protein, where \(L_P^{\text{max}}\) is the maximum length of a protein sequence, which is set to 1000 in this study. This protein matrix \(P\) is fed into the first convolutional layer and convolved by the weights \(c_1 \in \mathbb{R}^{s_1 \times D_P}\), where \(s_1\) is the length of the filter. This operation is repeated \(m_1\) times with the same filter length. Then this first convolution layer produces a vector \(PC_1 \in \mathbb{R}^{L_P^{\text{max}} - s_1 + 1}\), where elements in \(PC_1\) convey the \(s_1\)-gram features across the sequence. Multiple convolutional layers can be stacked on top of the previous output of the convolutional layer. After \(v\) number of convolution layers, the final vector, \(PC_1 \in \mathbb{R}^{(L_P^{\text{max}} - s_1 + s_2 - \cdots - s_v) \times m_v}\), is fed into the max pooling layer. This max pooling layer selects the most salient features from the vectors produced by the filters from the last layer. Then, the output of this max pooling layer is a vector \(P_{\text{rep}} \in \mathbb{R}^{D_P}\) \((m_v = D_P)\).

4.4.4 Interaction Denses

A molecule representation \((M_{\text{rep}} \in \mathbb{R}^{D_M}\), Section 4.4.2) and a protein representation \((P_{\text{rep}} \in \mathbb{R}^{D_P}\), Section 4.4.3) are concatenated to create the input of Interaction Denses, \(MP_{\text{rep}} \in \mathbb{R}^{D_M + D_P}\). Interaction Denses approximates the affinity score through a multi-layered feed-forward network with dropout regularization. The final

\(^4\)Adding an embedding layer slightly improves the accuracy of the DTI model.
Table 4.1: Statistics of the Davis and Kiba datasets. TRN/DEV/TST: training, development, evaluation sets.

<table>
<thead>
<tr>
<th>Dataset</th>
<th># of Compounds</th>
<th># of Proteins</th>
<th># of Interactions</th>
<th>TRN</th>
<th>DEV</th>
<th>TST</th>
</tr>
</thead>
<tbody>
<tr>
<td>DAVIS</td>
<td>68</td>
<td>442</td>
<td>30,056</td>
<td>20,037</td>
<td>5,009</td>
<td>5,010</td>
</tr>
<tr>
<td>KIBA</td>
<td>2,111</td>
<td>229</td>
<td>118,254</td>
<td>78,836</td>
<td>19,709</td>
<td>19,709</td>
</tr>
</tbody>
</table>

4.5 Experiments

4.5.1 Datasets

Drug-Target Interaction

The proposed MT-DTI model is evaluated on two benchmarks, Kiba [152] and Davis [35], because they have been used for evaluation in previous drug-target interaction studies [116, 66, 114]. Davis is a dataset comprised of large-scale biochemical selectivity assays for clinically relevant kinase inhibitors with their respective dissociation constant ($K_d$) values. The original $K_d$ values are transformed into log space, $pK_d$, for numerical stability, as suggested by [66] as follows:

\[
pK_d = -\log_{10}(\frac{k_d}{1e9})
\]

While Davis measures a bioactivity from one source of score, $K_d$, Kiba combines heterogeneous scores, $K_i$, $K_d$ and $IC_{50}$ by optimizing consistency among them. SimBoost [66] filtered out proteins and compounds with less than 10 interactions for computational efficiency, and we follow this procedure for a fair comparison. The number of compounds, proteins and interactions of the two datasets are summarized in Table 4.1. To facilitate comparison and reproducibility, we followed the same 5-fold
cross validation sets with a held-out test set which is publicly available\(^5\).

**Pre-training Dataset**

We downloaded the chemical compound information from the PubChem database \(^5\)\(^6\). Only canonical SMILES information were used to maintain consistency of representation. A total of 97,092,853 molecules are available in the canonical SMILES format.

**Drugbank Database**

The DrugBank database comprises a bioinformatics and cheminformatics resource that provides known drug-target interaction pairs. To prove the effectiveness of drug candidates generated by our model, we designed a case study (Section 4.6.1) using this database. We extracted 1,794 drugs from the database, excluding any compounds that were used when training our model. These selected compounds were the input to the trained model (by Kiba dataset) along with a specific protein to generate corresponding Kiba scores. The scores were used to find the best candidate drugs targeting that protein.

**4.5.2 Training Details**

Molecule Transformer is first trained with the collected compounds from the PubChem database (Section 4.5.1), and then the trained Transformer is plugged into the MT-DTI model for fine-tuning.

**Pre-training**

We use 97 million molecules for pre-training. Before feeding it to the Molecule Transformer, we tokenize each molecule at the character level. If the length of the molecules

\(^5\)https://github.com/hkmztrk/DeepDTA/

is more than 100, we truncate its head and tail together to have a fixed size of 100. We choose the middle part of the longer sequence so that the model can easily distinguish truncated sequences by simply looking at the existence of [BEGIN] and [END] tokens. The network structure of the Molecule Transformer is as follows. The number of layers is 8, the number of heads is 8, the hidden vector size is 128, the intermediate vector size is 512, the drop-out rate is 0.1, and the activation is Gelu. These parameters are picked from preliminary experiments and the hyperparameters used in the NLP model, BERT. We hypothesized that finding a chemical structure might be roughly 2-4 times easier task than finding a language model, because the size of the SMILES vocabulary is smaller than natural languages (70 vs 30k). Although the SMILES vocabulary is 400 times simpler, the number of tokens in the PubChem molecule datasets is about 2.4 times more than what BERT used to pre-train (8B vs 3.3B). This indicated that the molecules might have more complexity than expected when only considering the size of the vocabulary. Therefore we used parameters that were 2-4 times smaller than BERT. We note that there may be other parameter sets that can yield even better performance. We use the batch size of 512 and the maximum token size of 100, which enables it to process 50K tokens in one batch. Considering the average length of the compound sequence is around 80, there are approximately 8 billion tokens in the training corpus. We pre-train Molecule Transformer for 6.4M steps, which is equivalent to 40 epochs (8B/50K*40=6.4M). With an 8-core TPU machine, the pre-training took about 58 hours. The final accuracy of the Masked LM task was about 0.9727, which is comparable to the 0.9855 achieved by BERT on natural language.

Fine-tuning

The specifications of the Molecule Transformer in the MT-DTI model are the same as the one used when pre-training (Section 4.5.2). The Protein CNNs (Section 4.4.3)
consists of one embedding layer, three CNN layers, and one max pooling layer. It uses 128-dimensional vectors for the embedding layer. For CNN blocks, we denote the filter size as $K$ and the number of the filter as $L$. The final model parameter settings of CNNs are $K_1, K_2, K_3 = 12$ (Kiba), 8 (Davis) and $L_1 = 32, L_2 = 64, L_3 = 96$. The max pooling layer selects the best token representations from the last CNN layer, which makes the feature length as 96. Interaction Dense (Section 4.4.4) is comprised of three feed-forward layers and one regression layer. The layer sizes, when training Kiba, are 1024, 1024, 512 in order of the feature input to the regression layer and the learning rate, $\gamma$, is 0.0001. We reduce the network complexity when training Davis due to the small number of training samples. We use two feed-forward layers of sizes 1024 and 512. The learning rate is adjusted to 0.001. The entire network uses the same dropout rate of 0.1. All the hyper-parameters are tuned based on the lowest mean square error of the development sets for each fold, and the final score is evaluated on the held-out test set with the model at 1000 epochs.

4.5.3 Evaluation Metrics

We use four metrics to evaluate the proposed model: mean squared Error (MSE), concordance index (CI) [59], $r^2_m$, and area under the precision-recall curve (AUPR). MSE is a typical loss in the optimizer. CI is the probability that the predicted scores of two randomly chosen drug-target pairs, $y_i$ and $y_j$, are in the correct order:

$$CI = \frac{1}{N} \sum_{y_i > y_j} h(\hat{y}_i > \hat{y}_j),$$
where \( N \) is a normalization constant (the number of data pairs) and \( h(\cdot) \) is a step function [114]:

\[
h(x) = \begin{cases} 
1, & x > 0 \\
0.5, & x = 0 \\
0, & \text{else}
\end{cases}
\]

The \( r^2_m \) [125, 135] index is a metric for quantitative structure-activity relationship models (QSAR models). Mathematically,

\[
r^2_m = r^2 \times (1 - \sqrt{r^2 - r^2_0}),
\]

where \( r^2 \) and \( r^2_0 \) are the squared correlation coefficients with and without intercept, respectively. An acceptable model should produce an \( r^2_m \) value greater than 0.5.

Since AUPR is a metric for binary classification, we transform the regression scores to binary labels using known threshold values [66, 152]. For Davis, pairs with \( pK_d \geq 7 \) are marked as binding (1), others as no binding (0), and for Kiba, pairs with KIBA score \( \geq 12.1 \) are marked as binding (1), others as no binding (0).

4.5.4 Baselines

For the baseline methods, two similarity-based models and one deep learning-based model, the current SOTA, are tested. One of the similarity-based models is KronRLS [116], whose goal is to minimize a typical squared error loss function with a special regularization term. The regularization term is given as a norm of the prediction model, which is associated with a symmetric similarity measure. Another similarity-based model is Simboost [66], which is based on a gradient boosting machine. Simboost utilizes many kinds of engineered features, such as network metrics, neighbor statistics, PageRank [115] scores, and latent vectors from matrix factorization. The last one is a deep learning model, which is the SOTA method in predicting
Table 4.2: Test set results of the proposed MT-DTI model, MT-DTI model without fine-tuning (denoted as MT-DTI\textsuperscript{w/oFT}), and other existing approaches.

<table>
<thead>
<tr>
<th>Datasets</th>
<th>Method</th>
<th>CI (std)</th>
<th>MSE</th>
<th>( r_m^2 ) (std)</th>
<th>AUPR (std)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kiba</td>
<td>KronRLS</td>
<td>0.782 (0.001)</td>
<td>0.411</td>
<td>0.342 (0.001)</td>
<td>0.635 (0.004)</td>
</tr>
<tr>
<td></td>
<td>SimBoost</td>
<td>0.836 (0.001)</td>
<td>0.222</td>
<td>0.629 (0.007)</td>
<td>0.760 (0.003)</td>
</tr>
<tr>
<td></td>
<td>DeepDTA</td>
<td>0.863 (0.002)</td>
<td>0.194</td>
<td>0.673 (0.009)</td>
<td>0.788 (0.004)</td>
</tr>
<tr>
<td></td>
<td>MT-DTI\textsuperscript{w/oFT}</td>
<td>0.844 (0.001)</td>
<td>0.220</td>
<td>0.584 (0.002)</td>
<td>0.789 (0.004)</td>
</tr>
<tr>
<td></td>
<td>MT-DTI</td>
<td>\textbf{0.882 (0.001)}</td>
<td>\textbf{0.152}</td>
<td>\textbf{0.738 (0.006)}</td>
<td>\textbf{0.837 (0.003)}</td>
</tr>
<tr>
<td>Davis</td>
<td>KronRLS</td>
<td>0.871 (0.001)</td>
<td>0.379</td>
<td>0.407 (0.005)</td>
<td>0.661 (0.010)</td>
</tr>
<tr>
<td></td>
<td>SimBoost</td>
<td>0.872 (0.002)</td>
<td>0.282</td>
<td>0.644 (0.006)</td>
<td>0.709 (0.008)</td>
</tr>
<tr>
<td></td>
<td>DeepDTA</td>
<td>0.878 (0.004)</td>
<td>0.261</td>
<td>0.630 (0.017)</td>
<td>0.714 (0.010)</td>
</tr>
<tr>
<td></td>
<td>MT-DTI\textsuperscript{w/oFT}</td>
<td>0.875 (0.001)</td>
<td>0.268</td>
<td>0.633 (0.013)</td>
<td>0.700 (0.011)</td>
</tr>
<tr>
<td></td>
<td>MT-DTI</td>
<td>\textbf{0.887 (0.003)}</td>
<td>\textbf{0.245}</td>
<td>\textbf{0.665 (0.014)}</td>
<td>\textbf{0.730 (0.014)}</td>
</tr>
</tbody>
</table>

drug-target interactions, called DeepDTA [114]. It is an end-to-end model that takes a pair of sequences, (molecule, protein), and directly predicts affinity scores from the model. Features are automatically captured through back propagation of the multi-layered convolutional neural networks.

### 4.5.5 Results

The comparisons of our proposed MT-DTI model to the previous approaches are shown in Table 4.2. Reported scores are measured on the held-out test set using five models trained with the five different training sets. The best model parameters are selected based on the development set scores. MT-DTI outperforms all the other methods in all of the four metrics. The performance improvement is more noticeable when there are many training data where the improvements of Kiba are 0.019, 0.042, 0.065, and 0.04 compared with Davis’s improvements of 0.009, 0.016, 0.035, and 0.016, for CI, MSE, \( r_m^2 \), and AUPR, respectively. Furthermore, our model tends to be more stable with a larger training set, with the lowest standard deviation for CI and AUPR. Another interesting point is that our method without fine-tuning (MT-DTI\textsuperscript{w/oFT} in Table 4.2) produced competitive results. It outperforms the similarity based metrics and performs better than Deep-DTA for some metrics. This suggests
that the molecule representation using pre-training learns some useful chemical structure that can be exploited by the interaction denses model.

4.6 Case Studies

4.6.1 Anticancer Drug Discovery

We performed a case study using actual FDA-approved drugs targeting a specific protein, epidermal growth factor receptor (EGFR). This protein is chosen because this is one of the famous genes related to many cancer types. We calculated the interaction scores between EGFR and the 1,794 selected molecules based on the DrugBank database (see Section 4.5.1 for the details). These scores are sorted in descending order and summarized in Table 4.3.

EGFR is a transmembrane protein that is activated by binding of ligands such as epidermal growth factor (EGF) and transforming growth factor alpha (TGFα) [68]. Mutations in the coding regions of the EGFR gene are associated with many cancers, including lung adenocarcinoma [145]. Several tyrosine kinase inhibitors (TKIs) have been developed for the EGFR protein, including gefitinib, erlotinib, and afatinib. More recently, Osimertinib was developed as a third generation TKI targeting the T790M mutation in the exon of the EGFR gene [149]. Since the direct binding of these drugs to EGFR protein is well known, we tested whether our proposed model can identify known drugs for the EGFR protein.

Biological Insights

The result indicated that our model successfully identified known EGFR targeted drugs as well as novel chemical compounds that were not reported for association with the EGFR protein. For example, the first and second generation TKIs, such as Erlotinib and Gefitinib, and Afatinib, respectively, were predicted to exhibit high
<table>
<thead>
<tr>
<th>Ranking</th>
<th>Compound ID</th>
<th>Compound Name</th>
<th>KIBA Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>208908</td>
<td>Lapatinib*</td>
<td>14.002403</td>
</tr>
<tr>
<td>2</td>
<td>11557040</td>
<td>Lapatinib Ditosylate*</td>
<td>13.811217</td>
</tr>
<tr>
<td>3</td>
<td>10184653</td>
<td>Afatinib*</td>
<td>13.404812</td>
</tr>
<tr>
<td>4</td>
<td>16147</td>
<td>Triamcinolone Acetonide Sodium Phosphate</td>
<td>13.147043</td>
</tr>
<tr>
<td>5</td>
<td>5485201</td>
<td>Naltrexone Hydrochloride</td>
<td>13.114577</td>
</tr>
<tr>
<td>6</td>
<td>123631</td>
<td>Gefitinib*</td>
<td>13.111686</td>
</tr>
<tr>
<td>7</td>
<td>60699</td>
<td>Topotecan Hydrochloride</td>
<td>13.108758</td>
</tr>
<tr>
<td>8</td>
<td>5360515</td>
<td>Naltrexone</td>
<td>13.065864</td>
</tr>
<tr>
<td>9</td>
<td>441351</td>
<td>Rocuronium Bromide</td>
<td>13.032806</td>
</tr>
<tr>
<td>10</td>
<td>6918543</td>
<td>Almitrine Mesylate</td>
<td>13.016999</td>
</tr>
<tr>
<td>11</td>
<td>176870</td>
<td>Erlotinib*</td>
<td>12.885199</td>
</tr>
<tr>
<td>12</td>
<td>23422</td>
<td>Tubocurarine Chloride Pentahydrate</td>
<td>12.87076</td>
</tr>
<tr>
<td>13</td>
<td>6000</td>
<td>Tubocurarine</td>
<td>12.809549</td>
</tr>
<tr>
<td>14</td>
<td>11954379</td>
<td>Erlotinib Variant*</td>
<td>12.782704</td>
</tr>
<tr>
<td>15</td>
<td>11954378</td>
<td>Erlotinib Hydrochloride*</td>
<td>12.768639</td>
</tr>
<tr>
<td>16</td>
<td>3389</td>
<td>Prolxin Enanthate</td>
<td>12.737285</td>
</tr>
<tr>
<td>17</td>
<td>23724988</td>
<td>Oxycodone Hydrochloride Trihydrate</td>
<td>12.709352</td>
</tr>
<tr>
<td>18</td>
<td>14676</td>
<td>Methdilazine Hydrochloride</td>
<td>12.662965</td>
</tr>
<tr>
<td>19</td>
<td>5281065</td>
<td>Ibutilide Fumarate</td>
<td>12.650397</td>
</tr>
<tr>
<td>20</td>
<td>9869929</td>
<td>Avanafil</td>
<td>12.635439</td>
</tr>
<tr>
<td>21</td>
<td>60700</td>
<td>Topotecan</td>
<td>12.618897</td>
</tr>
<tr>
<td>22</td>
<td>5360733</td>
<td>Nalbufine Hydrochloride</td>
<td>12.610958</td>
</tr>
<tr>
<td>23</td>
<td>5282487</td>
<td>Paroxetine Hydrochloride Hemihydrate</td>
<td>12.608804</td>
</tr>
<tr>
<td>24</td>
<td>66259</td>
<td>Oxymetazoline Hydrochloride</td>
<td>12.557486</td>
</tr>
<tr>
<td>25</td>
<td>5311066</td>
<td>Desonide</td>
<td>12.538858</td>
</tr>
<tr>
<td>26</td>
<td>2247</td>
<td>Astemizole</td>
<td>12.536284</td>
</tr>
<tr>
<td>27</td>
<td>11954293</td>
<td>Asenapine</td>
<td>12.534941</td>
</tr>
<tr>
<td>28</td>
<td>11304743</td>
<td>Riociguat</td>
<td>12.527533</td>
</tr>
<tr>
<td>29</td>
<td>82153</td>
<td>Flunisolide</td>
<td>12.527164</td>
</tr>
<tr>
<td>30</td>
<td>71496458</td>
<td>Osimertinib*</td>
<td>12.507524</td>
</tr>
</tbody>
</table>

Table 4.3: Compound ranking based on the predicted Kiba scores when the target is EGFR protein. All compounds are from Drugbank database excluded any compounds in Kiba dataset. [Compound Name]* represents a known EGFR targetting drug.
affinity to the EGFR protein (Table 3). Lapatinib [104], which inhibits the tyrosine kinase activity associated with two oncogenes, EGFR and HER2/neu (human EGFR type 2), was predicted to exhibit the highest affinity. Osimertinib was also identified. Interestingly, chemical compounds targeting opioid receptors (naloxone hydrochloride, nalbuphine hydrochloride, and oxycodone hydrochloride trihydrate) for pain relief, antihistamines (methdilazine hydrochloride and astemizole), antipsychotic medication for schizophrenia (Prolixin Enanthate and Asenapine), and corticosteroids for skin problems (Triamcinolone acetonide sodium phosphate, Oxymetazoline hydrochloride, Desonide) were predicted to be associated with EGFR. Among these chemical compounds, Astemizole was suggested as a promising compound when treated with known drugs for lung cancer patients [47, 37]. Therefore, further investigations of these chemicals may provide a new therapeutic strategy for lung cancer patients.

4.6.2 Antiviral Drug Discovery

We performed another case study to suggest candidate molecules for the novel coronavirus found in Wuhan of China (COVID-2019). As the infection of COVID-2019 is rapidly spreading and there is a lack of effective treatment options for it, various strategies are being tested in many countries, including drug repurposing. In this case study, we used our MT-DTI model to identify commercially available drugs that could act on viral proteins of COVID-2019.

**Target Proteins:** We first narrow down potential target proteins that could inhibit the infection of the virus by referencing the previous coronavirus studies [88, 93, 20, 166, 170, 180, 91]. As a result, we extract the following six target sequences from the COVID-2019 whole genome sequence, from the National Center for Biotechnology Information (NCBI) database: 3C-like proteinase, RNA-dependent RNA polymerase, helicase, 3’-to-5’ exonuclease, endoRNAse, and 2’-O-ribose methyltransferase.
Small molecules | $K_d$ in nM
---|---
Atazanavir | 94.94
Remdesivir | 113.13
Efavirenz | 199.17
Ritonavir | 204.05
Dolutegravir | 336.91
Asunaprevir | 581.77
Ritonavir* | 609.02
Simeprevir | 826.24

Table 4.4: DTI prediction results of antiviral drugs available on markets against a COVID-2019 targeting 3C-like proteinase. Ritonavir is expressed in canonical and isomeric SMILES, and * indicates the isomeric SMILES of ritonavir.

**Training Data:** Since this case study is real world problem that requires a broader set of molecules and proteins, we pre-process and combine two bigdata sources in the literature: the Drug Target Common (DTC) database [153] and BindingDB [97] database. Three types of efficacy value, $K_i$, $K_d$, and $IC_{50}$ are integrated by a consistence-score-based averaging algorithm (13) to make the Pearson correlation score over 0.9 in terms of $K_i$, $K_d$, and $IC_{50}$. The MT-DTI model trained by these integrated dataset has the potential power to predict interactions between antiviral drugs and COVID-2019 proteins because it covers a wide variety of species and target proteins unlike DAVIS or KIBA that only include a small portion of human proteins.

**Result:** The COVID-2019 3C-like proteinase was predicted to bind with atazanavir ($K_d$ 94.94 nM), followed by remdesivir, efavirenz, ritonavir, and other antiviral drugs that have a predicted affinity of $K_d > 100$ nM potency (Table 4.4).

No other protease inhibitor among antiviral drugs was found in the $K_d < 1,000$ nM range. Although there is no real-world evidence about whether these drugs will act as predicted against COVID-2019 yet, some case studies have been identified. For example, a docking study of lopinavir along with other HIV proteinase inhibitors of the CoV proteinase (PDBID 1UK3) suggests atazanavir and ritonavir, which are listed in the present prediction results, may inhibit the CoV proteinase in line with
<table>
<thead>
<tr>
<th>Small molecules</th>
<th>$K_d$ in nM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Grazoprevir</td>
<td>8.69</td>
</tr>
<tr>
<td>Ganciclovir</td>
<td>11.91</td>
</tr>
<tr>
<td>Remdesivir</td>
<td>20.17</td>
</tr>
<tr>
<td>Atazanavir</td>
<td>21.83</td>
</tr>
<tr>
<td>Daclatasvir</td>
<td>23.31</td>
</tr>
<tr>
<td>Acyclovir</td>
<td>26.66</td>
</tr>
<tr>
<td>Etravirine</td>
<td>33.09</td>
</tr>
<tr>
<td>Entecavir</td>
<td>52.83</td>
</tr>
<tr>
<td>Efavirenz</td>
<td>76.70</td>
</tr>
<tr>
<td>Asunaprevir</td>
<td>78.36</td>
</tr>
<tr>
<td>Abacavir</td>
<td>131.51</td>
</tr>
<tr>
<td>Dolutegravir</td>
<td>150.15</td>
</tr>
<tr>
<td>Lomibuvir</td>
<td>280.96</td>
</tr>
<tr>
<td>Penciclovir</td>
<td>312.93</td>
</tr>
<tr>
<td>Trifluridine</td>
<td>315.79</td>
</tr>
<tr>
<td>Danoprevir</td>
<td>405.66</td>
</tr>
<tr>
<td>Ritonavir</td>
<td>624.30</td>
</tr>
<tr>
<td>Saquinavir</td>
<td>704.86</td>
</tr>
<tr>
<td>Raltegravir</td>
<td>832.25</td>
</tr>
<tr>
<td>Lamivudine</td>
<td>999.92</td>
</tr>
</tbody>
</table>

Table 4.5: DTI prediction results of antiviral drugs available on markets against a COVID-2019 targeting RNA polymerase.

the inhibitory potency of lopinavir [36]. According to the prediction, viral proteinase-targeting drugs were predicted to act more favorably on the viral replication process than viral proteinase through the DTI model (Table 4.5-4.9). The results include antiviral drugs other than proteinase inhibitors, such as guanosine analogues (e.g., acyclovir, ganciclovir, and penciclovir), reverse transcriptase inhibitors, and integrase inhibitors.

Among the prediction results, atazanavir was predicted to have a potential binding affinity to bind to RNA-dependent RNA polymerase ($K_d$ 21.83 nM), helicase ($K_d$ 25.92 nM), 3’-to-5’ exonuclease ($K_d$ 82.36 nM), 2’-O-ribose methyltransferase ($K_d$ of 390.67 nM), and endoRNAse ($K_d$ 50.32 nM), which suggests that all subunits of the COVID-19 replication complex may be inhibited simultaneously by atazanavir (Table 4.5-4.9).
<table>
<thead>
<tr>
<th>Small molecules</th>
<th>$K_d$ in nM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Remdesivir</td>
<td>6.48</td>
</tr>
<tr>
<td>Simeprevir</td>
<td>23.34</td>
</tr>
<tr>
<td>Atazanavir</td>
<td>25.92</td>
</tr>
<tr>
<td>Grazoprevir</td>
<td>26.28</td>
</tr>
<tr>
<td>Asunaprevir</td>
<td>28.20</td>
</tr>
<tr>
<td>Telaprevir</td>
<td>40.75</td>
</tr>
<tr>
<td>Ritonavir</td>
<td>41.60</td>
</tr>
<tr>
<td>Lopinavir</td>
<td>78.49</td>
</tr>
<tr>
<td>Darunavir</td>
<td>90.38</td>
</tr>
<tr>
<td>Ganciclovir</td>
<td>108.21</td>
</tr>
<tr>
<td>Penciclovir</td>
<td>129.41</td>
</tr>
<tr>
<td>Etravirine</td>
<td>175.50</td>
</tr>
<tr>
<td>Raltegravir</td>
<td>299.81</td>
</tr>
<tr>
<td>Dolutegravir</td>
<td>333.32</td>
</tr>
<tr>
<td>Nelfinavir</td>
<td>365.96</td>
</tr>
<tr>
<td>Indinavir</td>
<td>401.78</td>
</tr>
<tr>
<td>Efavirenz</td>
<td>412.86</td>
</tr>
<tr>
<td>Entecavir</td>
<td>452.78</td>
</tr>
<tr>
<td>Ritonavir*</td>
<td>462.20</td>
</tr>
<tr>
<td>Boceprevir</td>
<td>510.35</td>
</tr>
<tr>
<td>Lomibuvir</td>
<td>543.41</td>
</tr>
<tr>
<td>Acyclovir</td>
<td>661.76</td>
</tr>
</tbody>
</table>

Table 4.6: DTI prediction results of antiviral drugs available on markets against a COVID-2019 targeting helicase.
<table>
<thead>
<tr>
<th>Small molecules</th>
<th>$K_d$ in nM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Simeprevir</td>
<td>13.40</td>
</tr>
<tr>
<td>Efavirenz</td>
<td>39.55</td>
</tr>
<tr>
<td>Remdesivir</td>
<td>45.20</td>
</tr>
<tr>
<td>Danoprevir</td>
<td>49.26</td>
</tr>
<tr>
<td>Ganciclovir</td>
<td>56.29</td>
</tr>
<tr>
<td>Penciclovir</td>
<td>71.76</td>
</tr>
<tr>
<td>Atazanavir</td>
<td>82.36</td>
</tr>
<tr>
<td>Entecavir</td>
<td>82.78</td>
</tr>
<tr>
<td>Daclatasvir</td>
<td>110.47</td>
</tr>
<tr>
<td>Grazoprevir</td>
<td>111.90</td>
</tr>
<tr>
<td>Asunaprevir</td>
<td>117.26</td>
</tr>
<tr>
<td>Ritonavir</td>
<td>182.51</td>
</tr>
<tr>
<td>Lomibuvir</td>
<td>182.65</td>
</tr>
<tr>
<td>Darunavir</td>
<td>195.73</td>
</tr>
<tr>
<td>Raltegravir</td>
<td>306.99</td>
</tr>
<tr>
<td>Dolutegravir</td>
<td>326.89</td>
</tr>
<tr>
<td>Lopinavir</td>
<td>959.76</td>
</tr>
</tbody>
</table>

Table 4.7: DTI prediction results of antiviral drugs available on markets against COVID-2019 targeting 3’-to-5’ exonuclease.

<table>
<thead>
<tr>
<th>Small molecules</th>
<th>$K_d$ in nM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Efavirenz</td>
<td>34.19</td>
</tr>
<tr>
<td>Atazanavir</td>
<td>50.32</td>
</tr>
<tr>
<td>Remdesivir</td>
<td>70.27</td>
</tr>
<tr>
<td>Ritonavir</td>
<td>124.36</td>
</tr>
<tr>
<td>Danoprevir</td>
<td>235.15</td>
</tr>
<tr>
<td>Grazoprevir</td>
<td>277.87</td>
</tr>
<tr>
<td>Dolutegravir</td>
<td>349.63</td>
</tr>
<tr>
<td>Lomibuvir</td>
<td>398.81</td>
</tr>
<tr>
<td>Lopinavir</td>
<td>472.08</td>
</tr>
<tr>
<td>Darunavir</td>
<td>562.40</td>
</tr>
<tr>
<td>Nelfinavir</td>
<td>576.82</td>
</tr>
<tr>
<td>Telaprevir</td>
<td>618.11</td>
</tr>
<tr>
<td>Abacavir</td>
<td>619.79</td>
</tr>
<tr>
<td>Raltegravir</td>
<td>727.37</td>
</tr>
<tr>
<td>Boceprevir</td>
<td>891.62</td>
</tr>
</tbody>
</table>

Table 4.8: DTI prediction results of antiviral drugs available on markets against a COVID-2019 targeting endoRNAse.
<table>
<thead>
<tr>
<th>Small molecules</th>
<th>( K_d ) in nM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Remdesivir</td>
<td>134.39</td>
</tr>
<tr>
<td>Atazanavir</td>
<td>390.67</td>
</tr>
<tr>
<td>Efavirenz</td>
<td>423.00</td>
</tr>
<tr>
<td>Boceprevir</td>
<td>433.93</td>
</tr>
</tbody>
</table>

Table 4.9: DTI prediction results of antiviral drugs available on markets against a COVID-2019 targeting 2’-O-ribose methyltransferase.

Also, ganciclovir was predicted to bind to three subunits of the replication complex of the COVID-19: RNA-dependent RNA polymerase (\( K_d \) 11.91 nM), 3’-to-5’ exonuclease (\( K_d \) 56.29 nM), and RNA helicase (\( K_d \) 108.21 nM). Lopinavir and ritonavir, active materials of AbbVies Kaletra, both were predicted to have a potential affinity to COVID-19 helicase (Table 4.6) and are suggested as potential MERS therapeutics [38]. Recently, approximately $2 million worth of Kaletra doses were donated to China [70], and a previous clinical study of SARS by Chu et al. [26] may support this decision. Another anti-HIV drug, Prezcobix of Johnson & Johnson, which consists of darunavir and cobicistat, was to be sent to China [70], and darunavir is also predicted to have a \( K_d \) of 90.38 nM against COVID-19s helicase (Table 4.6). However, there was no current supporting literature found for darunavir to be used as a CoV therapeutic. Although remdesivir is not a FDA approved drug, its predicted potency to COVID-19 resulted as follows: against RNA-dependent RNA polymerase (\( K_d \) 20.17 nM), helicase (\( K_d \) 6.48 nM), 3’-to-5’ exonuclease (\( K_d \) 45.20 nM), 2’-O-ribose methyltransferase (\( K_d \) of 134.39 nM), and endoRNAse (\( K_d \) 70.27 nM).

### 4.7 Discussion

This paper proposes a new molecule representation using the self-attention mechanism, which is pre-trained using publicly available big data of compounds. The trained parameters are transferred to our DTI model (MT-DTI) so that it can be fine-tuned using two DTI benchmark data. Experimental results show that our model outper-
forms all other existing methods with respect to four evaluation metrics. Moreover, the first case study of finding drug candidates targeting a cancer protein (EGFR) shows that our method successfully enlists all of the existing EGFR drugs in top-30 promising candidates. In addition, the antiviral drug discovery case study shows that the proposed method can produce useful results in a very short time.

This suggests our DTI model could potentially yield low-cost drugs and provide personalized medicines. Our model can be further improved as the proposed attention mechanism is also applied to represent proteins. However, we didn’t explore this direction for two reasons. One reason is that the length of a protein sequence is ten times longer than a molecule sequence on average, which takes a considerable amount of time for computation. Another reason is the need for a protein dataset which contains enough sufficient information to pre-train the model. Unfortunately, such a dataset is not readily available.

4.8 Contribution

In this project, my contributions are as follows.

- Designed the study with Dr. Ho.
- (solely) Developed the algorithm
- Ran experiments with Park
- Analyzed the results with Dr. Kang and Dr. Ho
Molecule Generation

In this chapter we present another way of molecule discovery, molecule generation. The proposed method combines a molecule generation model and a molecule optimization model. We submitted this work to Knowledge Discovery in Databases conference 2020.

5.1 Introduction

Drug discovery is an expensive process. According to DiMasi et al. [43], the estimated average cost to develop a new medicine and gain FDA approval is $1.4 billion. Among this amount, 40% of it is spent on the candidate compound generation step. In this step, around 5,000 to 10,000 molecules are generated as candidates but 99.9% of them will be eventually discarded and only 0.1% of them will be approved to the market. This inefficient nature of the candidate generation step serves as motivation to design an automated molecule search method. However, finding target molecules with the desired chemical properties is challenging because of two reasons. First, an efficient search is not possible because the search space is discrete to the input [85]. Second, the search space is too large that it reaches up to $10^{60}$ [124]. As such, this task is currently being tackled by chemistry and pharmaceutical experts and takes
years to design. Therefore, this study aims to accelerate the drug discovery process by proposing a deep-learning (DL) model that accomplishes this task effectively and quickly.

Recently, many methods of molecular design have been proposed [9, 138, 48, 57, 33, 89, 112, 63, 137, 175]. Among them, Matched Molecular Pair Analysis (MMPA) [62] and Variational Junction Tree Encoder-Decoder (VJTNN) [81] formulated molecular property optimization as a problem of molecular paraphrase. Just as a Natural Language Process (NLP) model produces paraphrased sentences, when a molecule comes in as an input to these models, another molecule with improved properties is generated by paraphrase. Although MMPA was the first to try this approach, it is not effective unless many rules are given to the model [81]. To mitigate this problem, Jin et al. [81] proposed VJTNN, an end-to-end molecule optimization model without the need for rules. By efficiently encoding and decoding a molecule with graphs and trees, it is the current state-of-the-art (SOTA) model for optimizing a single property (hereby referred to as a single-objective optimization task). However, it cannot optimize multiple properties at the same time (a multi-objective optimization task) because the model inherently optimizes only one property. As noted by Vogt et al. [162], Shanmugasundaram et al. [140], the actual drug discovery process frequently requires balancing of multiple compound properties.

With these motivations, we propose a new DL-based end-to-end model that can optimize multiple properties in one model. By extending the preceding problem formulation, we consider the molecular optimization task as a sequence-based controlled paraphrase (or translation) problem. The proposed model, controlled molecule generator (CMG), learns how to translate the input molecules given as sequences into new molecules as sequences that best reflect the properties of the molecules we want. Our model extends the Transformer model [160] that showed its effectiveness in machine translation. CMG encodes raw sequences through a deep network and decodes a new
molecule sequence by referencing that encoding and the desired properties. Since we represent the desired properties as a vector, this model inherently can consider multiple objectives simultaneously. Moreover, we present a novel loss function using pre-trained constraint networks to minimize generating invalid molecules. Lastly, we propose a novel beam search algorithm that incorporates these constraint networks into the beam search algorithm [105].

![Diagram](attachment:image.png)

(a) The overall architecture at training.

(b) The overall architecture at prediction.

Figure 5.1: The proposed controlled molecule generator model.

We evaluate the proposed model using two tasks (single-objective optimization and multi-objective optimization) and two analysis studies (case study and ablation study)\(^1\). We compare our model with six existing approaches including the current SOTA, VJTN. CMG outperforms all baseline models in both benchmarks. In addition, the our model shows the biggest diversity in the output molecule distribution.

\(^1\)Code and data are available at [https://anonymous.url](https://anonymous.url)
The case study demonstrates the practicality of our method through the target affinity optimization experiment using an actual experimental drug molecule. Lastly, the ablation study not only shows the effectiveness of each sub-part, but demonstrates the superiority of the proposed model itself without the sub-parts.

Contribution: The contributions of this paper are summarized as below;

- We formulate the multi-objective molecule optimization task as a sequence-based controlled molecule translation problem.
- We propose a new self-attention based molecule translation model that can reflect the multiple desired properties.
- We introduce new loss functions to incorporate the pre-trained constraint networks.
- We propose a novel beam search algorithm using the pre-trained constraint networks.
- We present how to curate appropriate training data to train the proposed model.

5.2 Related Work

Molecule property optimization: Molecule property optimization models can be divided into two types depending on the data representation: sequence representations and graph representations. One of the earlier approaches using sequence representations utilizes encoding rules [165], while the recent ones [57, 138, 89] are based on DL methods that learn to reconstruct the input molecule sequence. This is related to our work in terms of the input representation, but they offer subpar performance when compared to the SOTA models. Another group of research uses graph representations conveying structural information [32, 80, 136, 92, 34]. Among them, VJTN [81] and MMPA [62, 44, 34] are closely related to our work because
they formulate the molecule property optimization task as a molecule translation problem. From the model perspective, MMPA is a rule-based model and VJTNN is a supervised DL model. Although our approach is also based on a DL method, there is a big difference in practical use cases. A single VJTNN model is capable of optimizing a single property, while the proposed model can optimize multiple properties by using the controlled decoder. With these differences, we formulate the molecule property optimization task as a “controlled” molecule “sequence” translation problem. Other molecule generation methods include Junction Tree Variational Auto Encoder (JT-VAE) [79], Variational Sequence-to-Sequence (VSeq2Seq) [57, 5], Graph Convolutional Policy Network (GCPN) [178], and Molecule Deep Q-Networks (MolDQN) [184].

**Natural Language Generation Model:** Our model is inspired by the recent success in molecule representation using the self-attention technique [143]. By adopting the BERT [41] architecture to represent molecule sequences, their model becomes the SOTA in the drug-target interaction task. In terms of the model architecture, our work is related to Transformer [160] because we extend it to be applicable to the molecule optimization task. There is a controlled text generation model [72] in NLP domain. It is related to ours because they feed the desired text property as one of the inputs. However, all of these methods are designed for NLP tasks, therefore, they cannot be directly applied to molecule optimization tasks.

**Transfer learning:** DL-based transfer learning by pre-training has been applied to many fields such as computer vision [133, 54], NLP [71], speech recognition [75, 99], and health-care applications [141]. They are related to ours because we also pre-train the constrained networks and transfer the weights to the main model.
5.3 Problem Definition

- **Data**: Molecules in SMILES format
  - X: Molecule
  - pX: The property vector of the input molecule, X
  - Y: New molecule
  - pY: The property vector of the output molecule, Y

- **Task**: Produce a new molecule, Y, that best reflects pY by optimizing X

Given an input molecule X, its associated molecule property vector pX, and the desired property vector pY, the goal is to generate a new molecule Y with the property pY with the similarity of (X, Y) ≥ δ. Note that δ is a similarity threshold and the similarity measure is Tanimoto molecular similarity over Morgan fingerprints [131]. Formally, for two Morgan fingerprints, FX and FY, where both of them are binary vectors, the Tanimoto molecular similarity is defined as,

\[
sim(F_X, F_Y) = \frac{|F_X \cap F_Y|}{|F_X \cup F_Y|}
\]

5.4 Proposed Model: Controlled Molecule Generator

In this section, we introduce the proposed model, controlled molecule generator (CMG), for generating new molecules with user-specified desired properties and sim-
ilarity to an input molecule.

5.4.1 Model Overview

Our model extends the Transformer [160] to a molecular sequence by incorporating molecule properties and additional regularization networks. Inspired by the previous success in applying the self-attention mechanism to represent a molecule sequence [143], we treat each molecule just like a natural language sequence. As noted by [76], the context and structure information of atoms is important when understanding the properties of molecules, just as we use context and structure information when understanding natural language.

However, this NLP technique cannot be directly applied, because the structure of the molecular sequence differs from natural languages, where the hierarchy is a letter-word-sentence. Not only that, there is no training data available that is collected for the molecule translation task, while there are ample datasets in the NLP domain. To fill these gaps, we propose the controlled molecule generation model (Figure 5.1) and present how we gather the training data for this network (Section 5.5.1). We optimize the proposed model using three loss functions as briefly shown in Figure 5.1(a). In addition, we propose two constraint networks (Section 5.4.4 and Figure 5.3), including property prediction network (Figure 5.3(a)), and similarity prediction network (Figure 5.3(b)) to train the model more accurately. Lastly, we also present how we modify the beam search algorithm [105] to best exploit the existing auxiliary networks (Section 5.4.5), as briefly shown in Figure 5.1(b).

5.4.2 Background

To efficiently present the idea of the proposed model, we briefly overview Transformer [160], the basic building block of the proposed model.

**Input Embedding**: For a given input sequence, $X = \{x_1, x_2, \cdots, x_i, \cdots, x_L\}, x_i \in$
where $L$ is the length of the sequence and $V$ is the number of vocabulary, we transform each token into a continuous vector, which is the sum of a token embedding vector and the positional embedding vector. These token embeddings are similar to word embeddings [108] except they are randomly initialized, therefore, each token, $x_i$ is transformed into $v_i \in \mathbb{R}^d$, where $d$ is the token embedding size. The token embeddings themselves are not sufficient to represent a sequence with a self-attention network, because a self-attention doesn’t consider the sequence order when calculating the attention, unlike other attention mechanisms. Therefore, we add a fixed positional embedding, $p_i \in \mathbb{R}^d$, to $v_i$ that makes the final input representation, $e_i = v_i + p_i, e_i \in \mathbb{R}^d$.

Self-Attention Layer: These transformed vectors, $e_i$, are the inputs of the encoder, consisting of multiple stacks of a self-attention layer and a feed-forward network. Each self-attention layer possesses three dense networks; a query network ($f_{\theta_Q}, \theta_Q \in \mathbb{R}^{d \times h}$), key network ($f_{\theta_K}, \theta_K \in \mathbb{R}^{d \times h}$), and value network ($f_{\theta_V}, \theta_V \in \mathbb{R}^{d \times h}$), where $h$ is the hidden dimension. With these three networks, each input vector, $e_i$ is projected into three utility vectors, a query vector ($q_i$), key vector ($k_i$), and value vector ($v_i$). Now, the output of a self-attention layer is computed as:

$$S = \text{Attention}(Q, K, V)$$

$$= \text{softmax} \left( \frac{Q K^T}{\sqrt{h}} \right) V \in \mathbb{R}^{L \times h}$$  \hspace{1cm} (5.2)

This self-attention computation (Equation 5.2) can be repeated $H$ number of times with the same input, forming the multi-head attention.

Feed-Forward Layer: The outputs of this multi-head attention are concatenated and projected using another dense network, called an intermediate dense layer, which parameter is represented as $\theta_O \in \mathbb{R}^{H \cdot h \times d}$. Then, it forms the final output of one encoder block, $o_i \in \mathbb{R}^d$.

Encoder: The Transformer encoder is multiple stacks of the two layers; the self-
attention layer and the feed-forward layer explained above. Note that the sequence length is preserved because the self-attention is applied to its own sequence, which preserves the input-output length. Therefore, the final output of the Transformer encoder is \( z_i \in \mathbb{R}^d \), which is compatible to be an input to another encoder.

**Decoder**: The Transformer decoder includes not only the same sub-layers as the Transformer encoder but one additional layer, the cross attention layer. It is similar to the self-attention layer in that it’s controlled by three vectors, \((q_i, k_i, v_i)\). The difference is that its attention weights are calculated between the last unit’s output of the Transformer encoder and each unit of the decoder, while the self-attention layer calculates its weights between the same layers.

**Loss Function**: The output of the last unit of the Transformer decoder is passed through the two dense layers, one for producing logits for all tokens and another for producing vocabulary probabilities for all tokens, \( \hat{Y} = \{\hat{y}_1, \hat{y}_2, \cdots, \hat{y}_j, \cdots, \hat{y}_M\} \), where \( M \) is the number of the output tokens. Given an output sequence, \( Y = \{y_1, y_2, \cdots, y_j, \cdots, y_N\} \), and the predictions, \( \hat{Y} \), the loss function is a cross entropy that can be formally defined as:

\[
\mathcal{L}_T(\theta_T; X, p_X, p_Y) = -\frac{1}{N} \frac{1}{M} \sum_{n \in N} \sum_{j \in M} \sum_{v \in V} y_{v,j,n} \cdot \log(\hat{y}_{v,j,n})
\]

\( \theta_T \) denotes all parameters of the Transformer and \( N \) represents the number of training samples.

### 5.4.3 Molecule Translation Network

In the molecule translation network, two major modifications are applied to the Transformer model [160]. The first change is applied to the molecule embedding. The input molecule is represented by the simplified molecular-input line-entry system (SMILES) sequence [165]. It is comprised of characters representing atoms or structure indica-
To mark the beginning and the end of a sequence, we add “[BEGIN]” token at the first position of the sequence and “[END]” token at the last. Mathematically, a molecule is represented as $X_L = \{x_1, x_2, \ldots, x_L\}$, where $x_i$ could be either an atom or a structure indicator, and $L$ is the sequence length, which varies depending on a molecule. Since the input sequence is a series of characters without any spaces, we tokenize this sequence into a list of a single token, then pass these tokens to the Transformer encoder. Another modification is that we add chemical property awareness to the hidden layer of the Transformer model. We enrich token vectors of the last encoder by concatenating property vectors to each of the token vectors as shown in Figure 5.2. Formally, let $z_i$ be the token vectors in the last encoder. Then, the the new encoding vector becomes $z'_i = (z_i, p_X, p_Y) \in \mathbb{R}^{d + 2k}$, where $k$ represents the number of properties. Although it might be seen as a simple method, this empirically shows the best result among other types of configurations, such as property embeddings, disentangled encodings (property and non-property encodings), and concatenating property differential information instead of providing two raw vectors. The increased
vector size can be handled by adjusting the three weights of the projection network
($\theta_Q$, $\theta_K$, and $\theta_V$) in the last encoding layer.

**Curation of the Training Data:** Since CMG is based on sequence translation, we need to appropriately curate the dataset. Similar to the previous work [81], we gather molecule pair $(X, Y)$ so that $\text{similarity}(X, Y) \geq \delta$ from the available molecule list. Next, we calculate property scores $p_X$ and $p_Y$ using the third-party tool. The detailed information can be found in Section 5.5.

### 5.4.4 Constraint Networks

As described in Section 5.4.2, the cost function of the Transformer network (Equation 5.3) is the cross entropy between the target $(y_i)$ and predicted molecule $(\hat{y}_i)$. We hypothesize that this information is not enough to teach the generating model, because the error signals generated by that loss function can hardly capture the valuable information, such as if a predicted sequence pertains to the desired property or if it satisfies the similarity constraint. With this motivation, we add two constraint networks; the property prediction network (Section 5.4.4) and the similarity prediction network (Section 5.4.4).

**Property Prediction Network**

Since the model complexity of the Transformer is quadratic in the input length, additional networks should be as simple as possible and yet accurate. Therefore, we employ a single layer of Bidirectional Long Short Term Memory [69] (BiLSTM) network in the proposed constraint networks. As shown in Figure 5.3(a), the property prediction network (PropNet) takes the predicted molecule sequence $(\hat{y}_j)$ as an input. A left-to-right LSTM layer encodes an input vector $(\hat{y}_j)$ into a hidden vector, $\vec{h}_j \in \mathbb{R}^d$ by the following equations;
(a) Property Prediction Network. Gray indicates unused vectors.

(b) Similarity prediction network. The biLSTM layer is simplified because the details are described in (a). One biLSTM layer is being shared by two input sequences.

Figure 5.3: Two Constraint Networks.
\[ i_j = \sigma(W_i Y[:, j] + U_i h_{i-1} + b_i) \quad (5.4) \]
\[ f_j = \sigma(W_f Y[:, j] + U_f h_{i-1} + b_f) \quad (5.5) \]
\[ C_j = i_j \odot \tanh(W_C Y[:, j] + U_C h_{i-1} + b_c) + f_j \odot C_{j-1} \quad (5.6) \]
\[ o_j = \sigma(W_o Y[:, j] + U_o h_{i-1} + b_o) \quad (5.7) \]
\[ h_j = o_j \odot \tanh(C_j) \quad (5.8) \]

If we apply these same equations with different parameters on the opposite direction of the input, we get another hidden vector, \( \overrightarrow{h}_j \in \mathbb{R}^d \). We concatenate the last two vectors from each direction, \( \overrightarrow{h}_M \) and \( \overleftarrow{h}_M \), to create the property feature vector \( (h_{prop} = (\overrightarrow{h}_M, \overleftarrow{h}_M) \in \mathbb{R}^{2d}) \). This feature vector is fed into a dense network with two hidden layers. The output of the first dense layer has 100 neurons and the dimension of the next layer’s output is the same as the property dimension. Since the property values are not strictly from 0 to 1, we don’t apply any activation function to the last dense layer, while we apply the RELU activation on the first dense layer. Since we have the property prediction, \( \hat{p}_Y \) and the desired property \( (p_Y) \) from the input, we can create another loss function, which will enrich error signals by adding property awareness in predicting a molecule. The loss function is formally written as,

\[ L_P(\theta_T; X, p_X, p_Y) = \frac{1}{N} \sum_{n \in N} |p_{Y_n} - \hat{p}_{Y_n}|^2 \quad (5.9) \]

Note that the parameters of PropNet are pre-trained using all molecules in the training set. Since all properties can be calculated using a third-party library, property annotations can be automated. Once pre-trained, the parameters are transferred to the CMG network. When training the main network, we freeze the weights of PropNet such that its role in the main network is consistent.

**Implementation Details:** We use 64-dimensional hidden vectors in the biLSTM layer, and 100 dimensions in the second last dense layer. We use Adam optimizer [84]
with default parameters set by Tensorflow Keras\(^2\). The batch size is 4,096 and the number of epochs is 1,000. The best model was selected by evaluating 20\% of the data, the validation set of PropNet described above. As a result, we selected the model at the 715th epoch, the mean square error (MSE) on the test set is 0.08554. Considering the values of QED and DRD2 range from 0 to 1, and the values of penalized logp typically range from -10 to 10, this MSE is small enough to be used as a property estimation.

**Similarity Prediction Network**

Another constraint network is the similarity prediction network (SimNet). It takes two sequences as an input, one is a predicted molecule sequence \((\hat{y}_i)\) and the other one is the input molecule sequence \((x_i)\). Since one of the requirements of the model is to generate a molecule that is similar to input, we hypothesize that adding error signals to the loss function predicted by SimNet could be useful. We employ one layer of BiLSTM for SimNet, which is shared by two different inputs because two sequences need to have the same feature representation in order to be compared against each other. Applying the BiLSTM layer to a predicted molecule sequence will produce a feature vector of the predicted molecule, \(h_{\text{predicted}} \in \mathbb{R}^{2d}\). Likewise, the feature vector of the input molecule is \(h_{\text{input}} \in \mathbb{R}^{2d}\). We concatenate these two feature vectors as \((h_{\text{predicted}}, h_{\text{input}}) \in \mathbb{R}^{4d}\), so that the next two dense networks can capture the similarity between the two. The output of the first dense layer has 100 neurons with RELU activation, and the output of the next layer has one neuron with sigmoid activation because SimNet is designed for a binary classification task (similar or not). For the training dataset for SimNet, we annotate all training molecule pairs with either 1 (if similarity \(\geq \delta\)) or 0 (otherwise), by calculating the similarity using a third-party library. We let \(s_n\) be the binary label of the similarity between the two molecules.

\(^2\text{https://www.tensorflow.org/api_docs/python/tf/keras}\)
in \( n \)'th data, and \( \hat{s}_n \) be the predicted output of SimNet for the same input data. With these labels and the predictions of SimNet, we can create the last loss function, formally written as,

\[
\mathcal{L}_S(\theta_T; X, p_X, p_Y) = \frac{1}{N} \sum_{n \in N} s_n \log \hat{s}_n + (1 - s_n) \log (1 - \hat{s}_n) \tag{5.10}
\]

When training the whole model, we transfer the trained SimNet weights into the whole model and freeze the SimNet weights for the same reason as PropNet presented in Section 5.4.4. The SimNet network configuration is illustrated in Figure 5.3(b).

**Implementation Details**: We use 64-dimensional hidden vectors in the biLSTM layer, and 100 dimensions in the second last dense layer. We use the same optimizer as PropNet. The batch size is 4,096 and the number of epochs is 1,000. The best model was selected by evaluating 20% of the data, test set set of SimNet described above. As a result, we selected the model at the 755th epoch, which recorded the prediction accuracy of the test set as 97.59%.

The weights of these two constraint networks are transferred to the corresponding part in the main CMG model. These constraint networks in CMG are frozen when training CMG and predicting a new molecule using it.

**CMG Loss Function**

By combining Equations (5.3)-(5.10), we can obtain the CMG loss function:

\[
\mathcal{L}_{CMG} = \mathcal{L}_T + \lambda_p \mathcal{L}_P + \lambda_s \mathcal{L}_S, \tag{5.11}
\]

where \( \lambda_p \) and \( \lambda_s \) are weight parameters.

**5.4.5 Modified Beam Search with Constraint Networks**

When generating a sequence from CMG at testing, there is no gold output sequence that it can reference. Therefore we need to sequentially generate tokens until
Algorithm 2: Modified Beam Search

1: **Input:** Candidate molecules: $C_1, C_2, \cdots, C_b$, 
   Corresponding beam scores: $s_1, s_2, \cdots, s_b$
   Input molecule: $X$
   Desired property vector: $p_Y$
2: **for** $i = 1$ **to** $b$ **do**
3: \[ \hat{p}_i \leftarrow \text{PropNet}(C_i) \]
4: \[ p_d \leftarrow |p_Y - \hat{p}_i| \]
5: \[ s_{pn} \leftarrow \text{reduce\_mean}(1 - p_d) \]
6: \[ s_{sn} \leftarrow \text{SimNet}(X, C_i) \]
7: \[ s_i \leftarrow s_i + (s_{pn} + s_{sn}) \]
8: **end for**
9: best_index \(\leftarrow\) arg max $s_i$
10: **Output:** $C_{\text{best\_index}}$

we encounter the "[END]" token, like other sequence-based algorithms. For this process, a typical way is the beam search, where the model maintains top $b$ number of best candidate sequences when predicting each token. When all candidate sequences are complete and ready, the model outputs the best candidate in terms of a beam score, a cumulative log-likelihood score for a corresponding candidate. We hypothesize that other non-best candidates might be better than the chosen one in terms of the purpose of a task. For example, there is a possibility that low ranked molecules could be closer to the desired properties than the top molecule selected by the beam search.

Unlike a typical Transformer model, CMG has PropNet and SimNet, that can be used to evaluate candidates before naively selecting one according to beam scores. Therefore, we propose a modified beam search algorithm using our constraint networks as summarized in Algorithm 2. It runs with $b$ number of the completed candidates along with corresponding beam scores. For the property evaluation, we first get the predicted property of each candidate and get the absolute difference from the desired property (Line 3-4 in Algorithm 2). Since this difference is desired to be small, we calculate the property evaluation score ($s_{pn}$) by subtracting them from one
(Line 5 in Algorithm 2). The property could have multiple values, therefore, we take an average of all elements of this difference vector. For the similarity evaluation, we get the predicted similarity between the input $X$ and each candidate $C_i$ (Line 6 in Algorithm 2). Since we expect a candidate should be similar (label 1) to the input, we regard the predicted similarity as the raw score from SimNet. By adding these two predicted scores to the original beam scores, we obtain the modified beam scores (Line 7 in Algorithm 2). With this new score, we can select the best candidate (Line 9-10 in Algorithm 2).

5.4.6 Diversifying the Output

Unlike other variational models (VSeq2Seq and VJTNN), the proposed one encodes a fixed vector that is able to generate a single output for one input. In order to diversify the output for a fixed input, we re-parameterize the desired vector, $(p_1, p_2, p_3)$, as random variable by adding a Gaussian noise with a user-specified variance.

$$\tilde{p}_k \sim N(p_k, \sigma_k)$$  \hspace{1cm} (5.12)

For example, if the desired property vector is $(p_1, p_2, p_3)$, we feed $(p_1 + \alpha, p_2 + \beta, p_3 + \gamma)$, where $\alpha, \beta$ and $\gamma$ are samples drawn from $N(0, \sigma_1)$, $N(0, \sigma_2)$, and $N(0, \sigma_3)$.

5.5 Experiments

We compare the proposed model with state-of-the-art molecule optimization methods in the following tasks.

**Single Objective Optimization (SOO):** This task is to optimize an input molecule to have a better property while preserving a certain level of similarity between the input molecule and the optimized one. Since developing a new drug usually starts with an existing molecule [8], this task serves as a good benchmark.
Table 5.1: Molecules used in this study. About 54% of DRD2D molecules also belong to ZINC, therefore, the total number of molecules for this study is 260,939.

Multi-Objective Optimization (MOO): This task reflects a more practical scenario in drug discovery, where modifying an existing drug involves optimizing multiple properties simultaneously, such as similarity, lipophilicity scores, drug likeness scores, and target affinity scores. Since improving one property might often result in sacrificing other properties, this task is harder than a single-objective optimization task.

To present multi-faceted aspects of the proposed model, we additionally perform the following case studies.

Case Study: To evaluate the effectiveness of the proposed model, we present the result of an actual drug optimization task with an existing molecule in an experimental phase.

Ablation Study: For ablation study, we report the validity of the constraint networks both in training and testing phases.

5.5.1 Datasets

Training Set for CMG: As described in Table 5.1, we use the ZINC dataset and the DRD2 related molecule dataset (DRD2D) for our experiments. This is the same set of molecules as what Jin et al. used to evaluate their model (VJTNN). This set is chosen since Jin et al. set up the benchmarks with this data, one of which (single-objective optimization) is used in our study. From these 260k molecules, we exclude molecules that appear in the development and the test set of VJTNN, resulting 257,565 molecules. With these molecules, we construct training datasets by selecting molecule pairs \((X, Y)\) with the similarity is greater than or
equal to 0.4, following the same procedure in [81, 34]. The main difference from their curation processes is that we add all pairs of molecules even if any property is decreased. By doing this, we provide a more ample dataset to a deep model, so that it could be helpful in finding more useful patterns. As a result, the number of pairs in training data is significantly bigger than theirs. Among all possible pairs \((257K \times 257K = 67B)\), we select 10,827,615 pairs that satisfies similarity condition \((\geq 0.4)\). With the same similarity condition, Jin et al. [81] gathered less than 100K due to additional constraints of training sets, where the property values of each output molecule should be greater than the ones of the corresponding input. As previous related works [81, 89] did, we pre-calculate the following three chemical properties of all molecules in the training set:

- **Penalized logP (PlogP)** [89]: A measure of lipophilicity of a compound, specifically, the octanol/water partition coefficient (logP) penalized by the ring size and synthetic accessibility.

- **Drug likeness (QED)**: A measure of drug-likeness based on the quantitative estimate of drug-likeness proposed by Bickerton et al. [8]

- **Dopamine Receptor (DRD2)**: A measure of molecule activity against a biological target, the dopamine type 2 receptor.

**Training Set for PropNet**: Among 260,939 molecules, we excluded all molecules in the test sets of the two tasks; single-objective optimization, multi-objective optimization. The number of these remained molecules is 257,565. We construct the dataset for PropNet by arranging all molecules as inputs and the corresponding three properties as outputs. We randomly split this into the training and validation sets with a ratio of 8:2.

**Training Set for SimNet**: We use a subset of all 10,827,615 pairs in the CMG training set due to the simpler network configuration of SimNet. When sub-sampling
pairs, we tried to preserve the proportion of the similarity in the CMG dataset to best preserve the original data distribution. The reason behind this effort is that preserving the similarity distribution could possibly contribute to the SimNet accuracy although SimNet only uses binary labels. In addition, we try to preserve the similar/not-similar ratio to be about the same. By sampling about 10% of data, we gathered 997,773 number of pairs and the ratio of the positive samples is 49.45%. We randomly split this into the training and validation set with a ratio of 8:2.

All molecules are represented in the SMILES format, and we parse them into a single character. Since the number of possible characters in the SMILES format is 71 and there are two special tokens, the size of the vocabulary is 73 when parsing a molecule data into a sequence of vocabulary ids.

### 5.5.2 Pre-Training of Constraint Networks

We pre-train the two constraint networks using the training sets described in Section 5.5.1. The best models were selected by evaluating 20% of each dataset, the validation sets. The weights of these two constraint networks are transferred to the corresponding part in the main CMG model. These constraint networks in CMG are frozen when training CMG and predicting a new molecule using it.

### 5.5.3 Single Objective Optimization

The first task on which we evaluate the proposed model is the single objective optimization task proposed by Jin et al. [79]. The goal is to generate a new molecule with an improved PlogP score under the similarity constraint ($\delta = 0.4$). We used the same development and test sets provided by Jin et al. [79]. The number of data samples in the development set and the test set are 200 and 800, correspondingly.

**Baselines:** We compare the proposed method with the following baselines; MMPA, JT-VAE, GCPN, VSeq2Seq, MolDQN, and VJTNN introduced in Section 5.2.
Since Jin et al. [81] ran and reported almost all of the baseline methods on the single property optimization task (PlogP improvement task) with the same test sets, we cite their experiment results. For MolDQN, which is published after VJTNN, we referenced the scores from MolDQN paper [184].

Following the same experimental setup with the baselines [81], we generate 20 molecules for a single input molecule and record one valid molecule with the maximum PlogP improvement.

**Implementation Details:** We use 4 layers and 8 heads of self-attention and feed-forward layers for both the encoder and the decoder. The hidden vector size is 128 and the dimension of the intermediate dense layer is 256. We set the maximum sequence length to be 150 because the max length used in the previous self-attention based molecule representation model [143] was 100 and the default buffer size of a typical Transformer model is 50% of its maximum sequence length. For the two constraint networks, we use the same configuration as pre-training models of them so that they are compatible with each other when transferring the weights. We use Adam optimizer [84] with the learning rate=2.0, $\beta_1 = 0.9$ and $\beta_2 = 0.997$. We train the proposed model (CMG) using 10M of the training set (Section 5.5.1) for 500 epochs with a batch size of 4,096. The dimension of the property vector is three, where the first one is PlogP, the second one is QED, and the last one is DRD2 values. We use the desired property vector of $\{X_{P_{\log P}}, 0.0, 0.0\}$ with the sampled offset parameters of $\alpha = \{-1.0, -0.5, 0.0, 0.5, 1.0\}$, $\beta = \{0.1, 0.6\}$ and $\gamma = \{0.52, 0.8\}$.

**Metrics:** Since the task is to generate a molecule with an improved PlogP value, we measure an average of raw increments and its standard deviation among valid molecules with the similarity constraint met. Specifically, among 20 generated molecules for a given input, if none of them satisfies the similarity constraint ($\delta = 0.4$), then we score that sample as 0, otherwise, we choose one molecule with a maximum increment as VJTNN did [81]. After we repeat this for all 800 molecules in the test
<table>
<thead>
<tr>
<th>Method</th>
<th>Improvement</th>
<th>Diversity</th>
</tr>
</thead>
<tbody>
<tr>
<td>MMPA</td>
<td>3.29 ± 1.12</td>
<td>0.496</td>
</tr>
<tr>
<td>JT-VAE</td>
<td>1.03 ± 1.39</td>
<td>-</td>
</tr>
<tr>
<td>GCPN</td>
<td>2.49 ± 1.30</td>
<td>-</td>
</tr>
<tr>
<td>VSeq2Seq</td>
<td>3.37 ± 1.75</td>
<td>0.471</td>
</tr>
<tr>
<td>MolDQN</td>
<td>3.37 ± 1.62</td>
<td>-</td>
</tr>
<tr>
<td>VJTNN</td>
<td>3.55 ± 1.67</td>
<td>0.480</td>
</tr>
<tr>
<td>Proposed</td>
<td>3.92 ± 1.88</td>
<td>0.545</td>
</tr>
</tbody>
</table>

Table 5.2: Single objective optimization performance comparison on the penalized logP task. MolDQN results are from, and the scores of other baselines are from.

set, we report the average and standard deviation. In addition to the improvement score, we also measure the diversity defined by Jin et al. [81]. Although this diversity measure has been used by previous researches, it is limited in that it encourages the outputs to have low similarity around the threshold. However, if we need to generate diverse molecules around the similarity threshold, then this diversity measure can serve as the right metric.

The diversity measure is the average pairwise Tanimoto distance between the two molecules in each pair, where the distance is $\text{dist}(X, Y) = 1 - \text{sim}(X, Y)$. We measure the distances between an input molecule and validly generated molecules among 20 outputs. Then the final score is the average of them. With this metric, we can compare the diversity of learned output distributions.

**Result:** After we train the model using the training set described in Section 5.5.1, we generate new molecules by feeding inputs including desired chemical properties to the trained model. As discussed in Section 5.4.6, we add offsets to desired properties so that the output can be diversified. Since the number of generated samples for each input is set to 20, we use the desired property vector of $\{X_{\text{PlogP}}, 0.0, 0.0\}$ with a total of 20 combinations of $(\alpha, \beta, \gamma)$ that are sampled from the user-defined distributions. We select the best model using the development set, and the test set performance of that model is reported in Table 5.2. In the PlogP optimization task, the proposed model outperforms all baselines including the current SOTA, VJTNN, in terms of...
both the average improvement and the diversity by a large margin. Considering the two recently proposed methods (MolDQN and VJTNN) are competing in 0.18 difference, the proposed one surpasses the current SOTA by 0.37 improvement. The same trend can be found in the diversity comparison. The proportion of valid output molecules generated by CMG is approximately 90%, while VJTNN records 100%. This indicates that once CMG produces valid molecules, then their improvements are big enough to compensate for loss from the invalid molecules (about 10% of all outputs).

We also experimented with other single objective optimizations, QED and DRD2. Unlike the PlogP case, where CMG outperforms the previous methods, these cases show that CMG failed to outperform others. The reason stems from the distribution of the training set. The proportions of pairs with the QED and DRD2 improvement in the training set are just 5.9% and 0.08%, respectively. Therefore, when optimizing for QED or DRD2, the model would not fully extract the useful information from the training set. Since our model is trained once for all tasks (SOO and MOO), this small portion of information would potentially impact negatively for certain single property optimizations, such as QED and DRD2.

5.5.4 Multi Objective Optimization

The single objective optimization task has served as a standard benchmark in the deep-learning based molecule generation field [79, 178, 81]. However, the actual drug discovery process frequently requires balancing of multiple compound properties [162, 140]. Therefore we set up a new benchmark, multi-objective optimization (MOO). In this task, we jointly optimize three chemical properties for a given molecule. We set up the success criteria of the generated molecules in the MOO task as follows:

- $\text{sim}(X, Y) \geq 0.4$
### Table 5.3: Multi objective optimization performance comparison.

<table>
<thead>
<tr>
<th>Method</th>
<th>Success Rate All Samples (2365)</th>
<th>Sub Samples (50)</th>
</tr>
</thead>
<tbody>
<tr>
<td>VJTN</td>
<td>3.56%</td>
<td>4.00%</td>
</tr>
<tr>
<td>MolDQN</td>
<td>-</td>
<td>0.00%</td>
</tr>
<tr>
<td>Proposed</td>
<td>6.98%</td>
<td>6.00%</td>
</tr>
</tbody>
</table>

- PlogP improvement is at least 1.0
- QED value is at least 0.9
- DRD2 value is over 0.5

To create the development set of this task, we merge all three different development sets provided by VJTN, consisting of 1,038 molecules. Among those molecules, we exclude any molecules that already satisfy either the QED or DRD2 criteria. After this filtering, the final development set contains 985 molecules. We perform the same procedure for the test set, which reduces the number of molecules from 2,452 to 2,365.

**Baselines:** For this task, we include the top two baselines (MolDQN and VJTN) from the SOO task. While MolDQN can perform the MOO task by simply modifying the reward function, VJTN can’t perform as it is because it is designed for a single property optimization. Here are how we prepare those baselines for the MOO task.

- **MolDQN:** The reward function of MolDQN for this task is defined as

\[
r = \frac{1}{8}(\mathbb{1}(sim(X, Y) \geq 0.4)) + \frac{1}{8}(P \log P(Y) - P \log P(X) \geq 1.0) + \frac{1}{8}(QED \geq 0.9) + \frac{1}{8}(DRD2 > 0.5) + \frac{1}{8}\text{sim}(X, Y) + \frac{1}{8}P \log P(Y) + \frac{1}{8}QED(Y) + \frac{1}{8}DRD2(Y)
\]

The first four terms represent the exact goal of the task, and the last four terms provide continuous information about the goals. Since one MolDQN
model should be trained for each test sample and it requires significant time, evaluating all 2,365 samples requires more than three months with a 96-CPU server. Therefore, we sub-sample the test set into 50 while preserving the original distribution\(^3\) and we use it for the proxy evaluation of MolDQN. For each input, we generate 60 samples\(^4\) after training the model (with exploration rate set to zero) and report success if at least one of them satisfies the success criteria defined above.

- **VJTNN**: We sequentially optimize an input molecule using three trained models from VJTNN (models for PlogP, QED, and DRD2). Firstly, the PlogP model generates 20 molecules for an input molecule. We select the most similar molecules that satisfy PlogP criteria. Then, for this selected molecule, the QED model generates another 20 molecules that optimize QED values. We again select the most similar molecules that satisfy PlogP and QED criteria. Finally, the DRD2 model generates 20 molecules and we report success if any of them satisfies the success criteria.

**Implementation Details**: We use the same configuration and the same training sets used in the previous task, SOO. We select the best model based on the development set (985 samples). When predicting a new molecule, we sample 60 molecules per input for a fair comparison to the baselines. In this task, we use the desired property vector of \(\left\{ X_{P \log P}, 0.0, 0.0 \right\} \) and the sampled offset parameters, \(\alpha = \{1.0, 2.0, 3.0\} \), \(\beta = \{0.91, 0.94, 0.97, 1.0\} \), and \(\gamma = \{0.51, 0.6, 0.7, 0.8, 0.9\} \).

**Result**: We only compare VJTNN for all samples due to the infeasible running time of MolDQN as mentioned above. As Table 5.3 shows, the proposed model is

---

\(^3\)When sub-sampling, we tried to preserve the proportion of the PlogP values because other properties are already filtered by the corresponding thresholds. Not only that, to get unbiased samples, we vary random seeds and select one that the success rate of VJTNN and the proposed method can be approximately matched with the corresponding all-sample results.

\(^4\)We relaxed the 60-sample-condition and inspected all states (molecules) during the training because the success rate was 0% at the 50-test set. However, the success rate remains the same as 0% even if we inspect all molecules.
<table>
<thead>
<tr>
<th>Method</th>
<th>Success Rate</th>
<th>±</th>
</tr>
</thead>
<tbody>
<tr>
<td>VJTNN</td>
<td>3.56</td>
<td>-3.42</td>
</tr>
<tr>
<td>PNet</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SNet</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MBS</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Proposed</td>
<td></td>
<td></td>
</tr>
<tr>
<td>∅ ∅ ∅</td>
<td>6.98</td>
<td>-</td>
</tr>
<tr>
<td>∅ ∅ □</td>
<td>6.72</td>
<td>-0.26</td>
</tr>
<tr>
<td>□ ∅ ∅</td>
<td>6.77</td>
<td>-0.21</td>
</tr>
<tr>
<td>∅ □ ∅</td>
<td>6.26</td>
<td>-0.72</td>
</tr>
<tr>
<td>□ □ ∅</td>
<td>5.33</td>
<td>-1.65</td>
</tr>
<tr>
<td>□ □ □</td>
<td>5.33</td>
<td>-1.65</td>
</tr>
</tbody>
</table>

Table 5.4: Ablation study. MBS is modified beam search, PNet is PropNet, and SNet is SimNet.

almost two times more successful in this task. The sub-sample experiment shows similar performance for VJTNN and ours, while MolDQN is not able to generate any successful samples.

### 5.5.5 Ablation Study

To illustrate the effect of the two constraint networks and the modified beam search, we present the result of the ablation study in Table 5.4. We use the MOO task for this comparison, and the result of VJTNN is also included for the reference. It’s worthwhile to note that the proposed model without any constraint networks and the modified beam search still outperforms VJTNN by 1.77% point. The component with the biggest contribution is SimNet that improves the performance by 0.72% point from the model without it. Another interesting thing is the success rates of the last two models in Table 5.4 are identical. The possible explanation is that if a model is trained without any constraint networks, the neurons generating candidate molecules could not properly convey any information about similarity and properties that can be exploited in the modified beam search.
Figure 5.4: A case study: The Aniracetam optimization task to improve DRD2 score. The molecule produced by the proposed model achieved the better DRD2 score than the molecule optimized by MolDQN.

5.6 Case Study

We performed a case study using an actual drug that is under the experimental stage targeting Dopamine D2 receptor (DRD2). From Drugbank, we first enlist all DRD2 targeting drugs that are in either experimental or investigational stages. Among these 28 drugs, we select the lowest DRD2 scored drug, named Aniracetam for this study. The goal is to improve DRD2 score with minimum perturbation of other properties.

**Baselines:** Since one VJTNN model optimizes one property, we just run the DRD2 VJTNN model trained by Jin et al. [81] by feeding Aniracetam. For MolDQN, the reward function becomes simpler as follows:

$$ r = \frac{1}{2} \mathbb{1}(\text{sim}(X,Y) \geq 0.4) + \frac{1}{2} \text{DRD2}(Y) $$

**Implementation Details:** We use the same configuration and the same training

---

6https://www.drugbank.ca/
7Aniracetam: COC1=CC=C(C=C1)C(=O)N1CCCC1=O
sets used in the two previous tasks, SOO and MOO. We select the best model based on the development set of DRD2 task provided by VJTNN. In this task, we use the desired property vector of \( \{ X_{P_{\log P}}, 0.0, 0.0 \} \) and the sampled offset parameters, \( \alpha = \{0.0\} \), \( \beta = \{-0.1, -0.05, 0.0, 0.05, 0.1\} \), and \( \gamma = \{0.6, 0.7, 0.8, 0.9\} \). We allow 20 generated samples for the proposed method and VJTNN, while we evaluate all states of MolDQN samples.

**Result:** In Figure 5.4, we compare the molecules generated by MolDQN\(^8\) and ours\(^9\), excluding the result of VJTNN, because VJTNN didn’t generate valid \( \text{sim}(X, Y) \geq 0.4 \) molecules. In terms of the predicted DRD2 scores, our molecule reached 0.77 whereas MolDQN’s molecule only recorded 0.03. For the other two properties which should be unchanged, our molecule seems to be stable with changes in PlogP by -0.35 and QED by -0.03 when compared with the MolDQN molecule that showed larger changes especially in PlogP. Although one case study cannot prove the general superiority of the proposed model, the proposed model consistently outperforms other baselines in all benchmarks (SOO, MOO, and the case study).

5.7 Discussion

This paper proposes a new controlled molecule generation model using the self-attention based molecule translation model and two constraint networks. We pre-train and transfer the weights of the two constraint networks so that they can effectively regulate the output molecules. Not only that, we present a new beam search algorithm using these networks. Experimental results show that the proposed model outperforms all other baseline approaches in both single-objective optimization and multi-objective optimization by a large margin. Moreover, the case study using an actual experimental drug shows the practicality of the proposed model. In the ablation

\[^8\text{MolDQN Molecule: C=C(c1ccc(OC)cc1)N1CCCC1}\
\[^9\text{Proposed Molecule: COc1ccccc1N1CCN(C2CC(C(=O)N3CCCC3=O)=C2c2ccccc2)CC1}\]
study, we present how each sub-unit contributes to model performance.

5.8 Contribution

In this project, my contributions are as follows.

- Designed the study with Dr. Ho.
- (solely) Developed the algorithm
- Ran experiments with Park
- Analyzed the results with Dr. Ho
6

Conclusion

The availability of well-curated biological and chemical datasets and appropriate computational power has motivated this dissertation work, deep learning approaches in computerized drug discovery. Although many machine learning based attempts have shown its potential compared to the conventional drug discovery methodology, they have been limited in modeling complex and large datasets effectively.

Inspired by many successes of deep learning based methods in other domains, this dissertation investigates the deep learning approaches for four sub-tasks in the drug discovery pipelines. In particular, we focus on two parts: target identification (Wx and CWx) and molecule discovery (MT-DTI and CMG).

Wx is a disease-related target identification model that selects a succinct set of potential target using an end-to-end feature selection neural network. Wx provides a simple way of identifying biological targets while previous methods rely on a long pipeline of machine learning methods and human labors. In addition to the simplicity of use, the results on various cancer classification tasks illustrate the superiority of the proposed method. Moreover, some of the genes we found using Wx matches those found to be important in recent studies.

CWx is a disease-related target identification model that is based on Wx to
predict patient prognosis with a small set of features. CWx outperforms the other methods by effectively training the algorithm from an easy to hard problem. In addition, CWx has a linear execution time to complete the feature selection steps depending on the number of samples. While the information theoretical-based feature selection algorithms take longer to finish the feature selection procedure. Not only that, CWx found genes have been confirmed that they contribute to the survival by other studies.

**MT-DTI** is a molecule transformer based drug target interaction prediction model that adapts and modifies the self-attention mechanism, which is pre-trained using publicly available big data of compounds. Although this big dataset is not directly related to the downstream tasks, the proposed pre-training is shown to be useful in DTI tasks. Experimental results show that our model outperforms all other existing methods with respect to four evaluation metrics. Moreover, the first case study of finding drug candidates targeting a cancer protein (EGFR) shows that our method successfully enlists all of the existing EGFR drugs in top-30 promising candidates. In addition, the antiviral drug discovery case study shows that the proposed method can produce useful results in a very short time.

**CMG** is a controlled molecule generation model that is based on the Transformer model with two constraint networks. CMG successfully adapts self-attention based translation model in the controlled molecule optimization task using a desired property vector. Not only that, we further improve the model by adding two contraint networks that predict necessary characteristics of the generated molecules. We also present a new beam search algorithm using these constraint networks. Experimental results show that the proposed model outperforms all other baseline approaches in both single-objective optimization and multi-objective optimization by a large margin. Moreover, the case study using an actual experimental drug shows the practicality of the proposed model.
Future Direction

There are several ways to improve and extend the proposed methods.

**Interpretable DTI**: In MT-DTI, we present the best DTI model, however, it is a black box model that doesn’t provide precise binding pocket information, which could be valuable in other tasks of the drug discovery. By successful employing an appropriate attention mechanism to the DTI model, we can develop better and interpretable DTI models.

**Protein representation**: This dissertation has only explored the molecule representation via deep networks. Therefore, one possible direction is a new protein representation using deep networks. We only used the amino acids sequence for a protein, however, it actually forms three dimensional structures which contain much more information than a sequence. If we can effectively represent a protein using a predicted 3D structure, then the performance of downstream tasks could be improved.

**Protein protein interaction**: Once we have the better protein representation, we can apply the new representation to the protein protein interaction (PPI) task. Since a protein itself can be served as a drug, if we can understand complex PPI among many protein, it could be beneficial in developing antibody based drug development.


[15] Dong-Sheng Cao, Shao Liu, Qing-Song Xu, Hong-Mei Lu, Jian-Hua Huang, Qian-Nan Hu, and Yi-Zeng Liang. Large-scale prediction of drug–target in-


[38] Emmie de Wit, Neeltje van Doremalen, Darryl Falzarano, and Vincent J Mun-


[53] Kyle Yingkai Gao, Achille Fokoue, Heng Luo, Arun Iyengar, Sanjoy Dey, and


[75] Navdeep Jaitly, Patrick Nguyen, Andrew Senior, and Vincent Vanhoucke. Application of pretrained deep neural networks to large vocabulary speech recog-


[120] Li Peng, Xiu Wu Bian, Chuan Xu, Guang Ming Wang, Qing You Xia, Qing Xiong, et al. Large-scale rna-seq transcriptome analysis of 4043 cancers and 548 normal tissue controls across 12 tcga cancer types. *Scientific reports*, 5:13413, 2015.


[130] Mark D Robinson, Davis J McCarthy, and Gordon K Smyth. edger: a bio-
conductor package for differential expression analysis of digital gene expression


An r package for omics feature selection and multiple data integration. *PLoS

[133] Rasmus Rothe, Radu Timofte, and Luc Van Gool. Dex: Deep expectation of
apparent age from a single image. In *Proceedings of the IEEE International

[134] Dimitrios H Roukos. Next-generation, genome sequencing-based biomarkers:

Kar, and Rudra Narayan Das. Some case studies on application of rm2 metrics
for judging quality of quantitative structure–activity relationship predictions:
emphasis on scaling of response data. *Journal of computational chemistry*, 34

[136] Bidisha Samanta, Abir De, Niloy Ganguly, and Manuel Gomez-Rodriguez. De-
signing random graph models using variational autoencoders with applications

[137] Benjamin Sanchez-Lengeling, Carlos Outeiral, Gabriel L Guimaraes, and Alán
Aspuru-Guzik. Optimizing distributions over molecular space. an objective-


[144] Bonggun Shin, Sungsoo Park, Keunsoo Kang, and Joyce C Ho. Self-attention


[179] Kun Yu, Kumaresan Ganesan, Lay Keng Tan, Mirtha Laban, Jeanie Wu, Xiao Dong Zhao, Hongmin Li, Carol Ho Wing Leung, Yansong Zhu, Chia Lin


