In Silico Methods for Screening Combination Drug Therapies

by

Allison Chelsea Tam

B.S., Massachusetts Institute of Technology (2019)

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of
Master of Engineering in Electrical Engineering and Computer Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

September 2020

© Massachusetts Institute of Technology 2020. All rights reserved.
In Silico Methods for Screening Combination Drug Therapies

by

Allison Chelsea Tam

Submitted to the Department of Electrical Engineering and Computer Science
on August 14, 2020, in partial fulfillment of the
requirements for the degree of
Master of Engineering in Electrical Engineering and Computer Science

Abstract

Drug cocktails are an effective tool for treating many complex diseases, such as cancer and AIDS. However, developing new combination therapies is challenging, due to the combinatorial nature of the search space. Machine learning offers an efficient way to screen for promising new combinations by learning from available experimental assays. Many such models have been proposed with no leading solution. Most importantly, existing methods fail to generalize to new combinations that involve compounds not seen in the training set, requiring prohibitively large datasets to produce meaningful results. In this thesis, we propose two data-efficient models and evaluate their performance on the NCI-ALMANAC dataset, which screens two-drug cocktails on various cancer cell lines. Our first model, ComboFiLM, significantly outperforms traditional descriptor-based models, especially on more imbalanced datasets. Our second model, Bundles, solely learns from single-agent data and can be applied zero-shot to combinations. Bundles matches the performance of an oracle that is trained on combination data, and significantly surpasses similarly pretrained models when finetuned on a fraction of available combination data.

Thesis Supervisor: Regina Barzilay
Title: Delta Electronics Professor
Acknowledgments

I am especially grateful to my advisor, Regina Barzilay, for welcoming me into her lab and giving me the opportunity to do research in an area that I have long been passionate about. Without her support and guidance, this work would not be possible. I would also like to thank Tommi Jaakkola, Wengong Jin, and Octavian-Eugen Ganea for their guidance and feedback. Our discussions were consistently some of the most illuminating and joyful parts of my week.

Thank you to Nithin Buduma, Kyle Swanson, Victor Quach, and Rachel Wu for being incredible labmates. Their insights and fun stories were an indispensable part of working in lab, even when we were physically apart. I would also like to thank Sidney Li, Nick Gomez, Magnus Johnson, Tugsbayasgalan Manlaibaatar, Claire Nord, and Anna Verkhovskaya, who have all been a source of strength and inspiration throughout this past year.

And finally, I would like to thank my parents and my brother for their unwavering love and support. I would not be the person I am today without them.
Contents

1 Introduction 13
  1.1 Existing Methods ................................. 14
  1.2 Our Approach .................................. 15
    1.2.1 ComboFiLM ............................... 16
    1.2.2 Disease Bundles ......................... 16
    1.2.3 Evaluation ............................... 18

2 Background 21

3 Methods 23
  3.1 ComboFiLM ................................... 23
    3.1.1 Learned Drug Representations ................. 24
    3.1.2 FiLM Layers ................................ 25
    3.1.3 Model Architecture ......................... 25
    3.1.4 Training Procedure ......................... 26
  3.2 Disease Bundles ................................ 27
    3.2.1 Model Framework .......................... 27
    3.2.2 Learning Latent Mechanisms ................. 31
    3.2.3 Training Procedure ......................... 32

4 Experimental Procedure 35
  4.1 Data ........................................... 35
  4.2 Evaluation Framework ........................... 38
4.2.1 Evaluating ComboFiLM ........................................... 39
4.2.2 Evaluating Disease Bundles ................................. 40

5 Results ......................................................................... 47
  5.1 ComboFiLM ............................................................... 47
  5.2 Disease Bundles ......................................................... 49
    5.2.1 Zero-Shot Setting ................................................. 49
    5.2.2 Data-Scarce Setting ............................................. 53
    5.2.3 Interpreting Bundle Vectors ................................. 55

6 Conclusion ..................................................................... 59

A Tables ........................................................................... 61

B Figures ........................................................................... 63
# List of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3-1</td>
<td>Illustration of ComboFiLM architecture.</td>
<td>26</td>
</tr>
<tr>
<td>3-2</td>
<td>Visualization of the predicted combination profile under the Bliss model Independence.</td>
<td>29</td>
</tr>
<tr>
<td>3-3</td>
<td>Illustration of how a disease bundle is trained to learn different latent mechanisms.</td>
<td>33</td>
</tr>
<tr>
<td>3-4</td>
<td>Illustration of how disease bundles are used to predict activity of a combination therapy.</td>
<td>34</td>
</tr>
<tr>
<td>4-1</td>
<td>Visualization of leave-out-compound split (left) and random split (right). Combination data can be viewed as a drug-drug matrix of labels.</td>
<td>39</td>
</tr>
<tr>
<td>4-2</td>
<td>Toy hierarchy of GO terms, as related to targets and compounds.</td>
<td>45</td>
</tr>
<tr>
<td>5-1</td>
<td>Boxplot of model performances on different cell lines. (ComboFiLM)</td>
<td>48</td>
</tr>
<tr>
<td>5-2</td>
<td>Relationship between model performance and hit rate of the dataset. (ComboFiLM)</td>
<td>49</td>
</tr>
<tr>
<td>5-3</td>
<td>Boxplot of relative model performances on different cell lines, as compared to Oracle. (Bundles)</td>
<td>50</td>
</tr>
<tr>
<td>5-4</td>
<td>Boxplot of overall and tiered performances on different cell lines. (Bundles)</td>
<td>51</td>
</tr>
<tr>
<td>5-5</td>
<td>Model performances varying with the percentage of dataset exposed to the model. (Bundles)</td>
<td>54</td>
</tr>
<tr>
<td>5-6</td>
<td>Distribution of radii for the MVEE’s demonstrating bundle collapse.</td>
<td>57</td>
</tr>
</tbody>
</table>
B-1 Distribution of linked SMILES per GO term in collected mechanism data. ................................................................. 63
B-2 Tiered model performances varying with the percentage of dataset exposed to the model. (Bundles) ........................................ 64
## List of Tables

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.1</td>
<td>Summary statistics per cell line in the NCI-ALMANAC dataset.</td>
<td>36</td>
</tr>
<tr>
<td>4.2</td>
<td>Tier-specific summary statistics per cell line in the NCI-ALMANAC dataset.</td>
<td>37</td>
</tr>
<tr>
<td>5.1</td>
<td>Performance of ComboFiLM compared to baselines. Mean AUPRC and standard deviation on 3-fold cross validation are reported for each cell line.</td>
<td>48</td>
</tr>
<tr>
<td>A.1</td>
<td>Ranges used for ComboFiLM hyperparameter searches.</td>
<td>61</td>
</tr>
<tr>
<td>A.2</td>
<td>Sizes of training, validation, test sets used in Bundles data-scarce setting.</td>
<td>61</td>
</tr>
<tr>
<td>A.3</td>
<td>Model performances in zero-shot setting.</td>
<td>62</td>
</tr>
<tr>
<td>A.4</td>
<td>Model performances in data-scarce setting after finetuning on the full combination dataset.</td>
<td>62</td>
</tr>
</tbody>
</table>
Chapter 1

Introduction

Drug cocktails are an important therapeutic tool. They are used to treat a variety of complex diseases, ranging from cancer to drug-resistant tuberculosis to acquired immunodeficiency syndrome (AIDS) [19, 1, 10]. Combination therapies may achieve greater efficacy and potency than monotherapies because of the potential for complementary drug actions. Drugs in combination can act at different points along the same pathway or affect different pathways altogether to help treat the disease. These pathway-level interactions uniquely contribute to the potential of a combination’s therapeutic activity [27].

We define a combination to be therapeutically active if it treats the disease. The ideal combination therapy is both active and synergistic. Synergy occurs when the observed effect is greater than the expected additive effect of two individual drugs. This can take the form of synergistic potency and/or synergistic efficacy. In the former, a drug becomes more potent in the presence of another, which reduces the required dose and the potential for side effects. In the latter, the maximal efficacy of the combination is greater than the therapeutic activity of the more effective single-agent drug [18].

Several classical synergy frameworks have been proposed over the years, such as Loewe additivity [15] and Chou-Talalay [6]. One of the more popular frameworks is Bliss independence, which assumes that drugs act independently. Under the Bliss model, their combined effect is estimated to be the probabilistic union of the two indi-
vidual drug effects [3]. However, these frameworks may sometimes produce conflicting definitions of synergy with no current gold standard.

Synergistic combinations, while ideal, are rare. Many therapeutically active combinations result from the additive effects of individual drugs. Yet, identifying the small number of active combinations remains a difficult task, because simply combining two drugs does not guarantee an increase in efficacy over the individual effects. An effective combination therapy ultimately needs to be active, not necessarily synergistic. Therefore, this thesis mainly focuses on modeling active combinations but provides insight into both activity and synergy.

1.1 Existing Methods

Historically, many approved cocktails have been the result of handcrafted trial-and-error experiments. Domain experts in different disease areas drive development by proposing combinations that target complementary pathways [2, 21]. With the rise of computational biology resources in recent years, researchers have started automating this process by analyzing protein-protein interaction networks [5].

While this biology-focused approach is labor and time expensive, an alternative direction is to use high-throughput screening (HTS) to systematically test and identify promising new combinations [4]. HTS, in conjunction with machine learning algorithms, has been successful when applied to single-agent drug development [26]. This approach, when applied to combination therapies, presents its own unique set of challenges—primarily the combinatorial explosion of the search space. Combinations must also be tested over a grid of concentrations in order to determine synergism, further exacerbating the problem. It is infeasible and expensive to systematically test all combinations constructed from large libraries of compounds.

Computational methods will have to adapt to combination datasets that often have worse class imbalance and cover a less diverse set of compounds than their single-agent counterparts. The ideal method must not only be data-efficient but also be able to generalize to new combinations that involve compounds not previously
seen in training (i.e. out-of-library combinations).

Existing methods predict combination activity or synergy by applying both feature engineering and standard machine learning techniques, such as random forest or deep neural networks. They characterize drugs with fixed molecular descriptors like Morgan fingerprints. Combinations are represented by concatenating drug descriptors and auxiliary features (e.g. disease genomics features, concentration information).

Learned drug representations, which are responsible for many significant advances in single-agent in silico screening, are not nearly as well studied when applied to combinations. Some methods have used graph convolutional networks on the graphs of drug structures to predict side effects [8]. However, side effect datasets are magnitudes larger than typical combination screens. Most importantly, none of these methods have demonstrated the ability to generalize to out-of-library combinations. This is a critical component that limits existing methods’ practical applicability to aid drug cocktail development.

### 1.2 Our Approach

In this thesis, we propose two new machine learning algorithms that can generalize to out-of-library combinations. They are the first models to utilize learned drug representations for predicting synergy or activity of combinations. Our first model, ComboFiLM, is trained on combination data to predict any joint property of a combination (e.g. therapeutic activity, synergy, side effects). It can be used for classification or regression tasks. Our second model, Bundles, is trained solely on single-agent activity data and can classify combinations as active or inactive in a zero-shot setting. The two models complement each other in their approaches. Together, they provide powerful prediction capability across all sizes of available combination datasets—even when there are none.
1.2.1 ComboFiLM

The ComboFiLM architecture comprises a message-passing neural network (MPNN) [31] followed by a classifier \( f \). The MPNN \( M \) encodes each drug in the combination, producing learned drug representations \( M(A) \) and \( M(B) \) for drugs \( A \) and \( B \). The classifier is a multilayer perceptron modulated by FiLM layers.

FiLM layers were first used in visual question-and-answering to integrate question information into visual processing [22]. As applied to drug cocktails, FiLM can be used to integrate information from the two drugs. In our classifier, the FiLM layer generates an affine transformation based on one drug and applies it to the output of the previous layer. The transformed vector is then passed to the next linear layer.

In effect, \( f(M(A), M(B)) \) computes the property of drug \( A \) in the context of drug \( B \), which is an asymmetric operation. The model restores commutative predictions by outputting the average of \( f(M(A), M(B)) \) and \( f(M(A), M(B)) \).

Compared to existing methods, ComboFiLM is a more powerful model. The MPNN allows the model to construct drug representations that would be specifically useful to the prediction task. Additionally, FiLM layers are capable of learning a superset of conditioning methods beyond concatenation-based conditioning. FiLM layers may also help regularize the MPNN by forcing the drug embeddings to be discriminative, as well as useful for conditioning.

1.2.2 Disease Bundles

*Bundles* is a zero-shot model for combinations. By primarily learning from single-agent data, it is meant to be used when combination data is unavailable or scarce. While ComboFiLM and other existing methods directly learn the relationship between drugs and activity, *Bundles* also learns the biological processes or “latent mechanisms” that drive therapeutic activity. For example, a latent mechanism for chemotherapy drugs could be inhibition of a critical target involved in DNA replication. Similar to how biologists propose combining drugs with complementary effects, our model learns and leverages these latent mechanisms to predict activity of combinations. Since data
limitations preclude us from directly learning the set of biological processes relevant to the disease, we induce these latent mechanisms to represent biological concepts during training.

Bundles consists of three components: drug representations, disease representations, and therapeutic activity prediction. The drug representations are produced by a MPNN [31]. The representation for a given disease consists of a “bundle” or a set of $K$ vectors. Each vector corresponds to a disease-specific latent mechanism. Given a disease and single-agent drug, our model predicts a $K$-dimensional single-agent profile. Each dimension is the predicted impact the drug has via a given latent mechanism, where impact is quantified by similarity between the bundle vector and drug representation. Given two single-agent profiles, we apply Bliss independence to each dimension to produce the combination profile. An active combination would have a profile with high impact in one or more latent mechanisms. To that end, the model predicts therapeutic activity to be the combination’s maximum impact of any one mechanism from the disease bundle.

From single-agent data, the latent mechanisms and single-agent drug representations are learned jointly. Several disease bundles can be learned at the same time. To encourage the bundle vectors within a disease to represent a diverse array of latent mechanisms, we use the following biological insight: similarly-structured compounds affect the body in similar ways. For example, antibiotics can generally be divided into different drug classes in which the molecular scaffold defines the mechanism of action (e.g. macrolides, fluoroquinolones, tetracyclines) [28].

During training, inactive single-agents are taught to have suppressed impact for each latent mechanism (i.e. have dissimilar representations). Meanwhile, active single-agents are divided into clusters using K-means on RDKit descriptors. These clusters keep similarly-structured compounds together and thus act as an approximation for mechanism. Each latent mechanism is learned jointly with the same pre-assigned cluster of active compounds throughout training. As training progresses, the flexibility of the MPNN helps overcome the imperfect prior provided by clusters.
1.2.3 Evaluation

We evaluate our models on their ability to generalize to out-of-library combinations from the NCI-ALAMANAC dataset. Our task is to identify cancer-inhibiting combinations for a given cell line. Our experiments demonstrate that ComboFiLM is relatively robust to class imbalanced datasets, achieving up to a 350% gain in AUPRC over descriptor-based models. Meanwhile, in a zero-shot setting, Bundles matches the performance of an MPNN-based oracle that is trained on combination data. When given access to any size of combination data, Bundles is more data-efficient. Furthermore, using drug-target data extracted from CHEMBL, we demonstrate that disease bundles can encode a diverse set of latent mechanisms that correlate to real biological processes. This underlying structure may be leveraged in the future to extend our model to predict synergy in a zero-shot setting.

Our work has two key contributions:

1. Compared to existing methods, we significantly expand the representational power of combination models, which in turn helps boost performance for identifying active combinations. Instead of descriptor-based models, we use learned drug representations that can be tuned to the specific disease or task. In one of our models, ComboFiLM, we allow the model to repeatedly refer back to the other drug representation throughout the classifier. This contrasts the drug concatenation input used by prior methods, which assumes where the model should require information from both compounds.

2. Incorporating the biology of combination therapies as inductive biases can significantly improve the data-efficiency of combination models. We use two main insights in Bundles. First, we explicitly learn the relationships between drugs, mechanisms, and diseases and leverage this understanding to identify promising combinations. This is the same traditional systems biology approach used by experts who propose combinations by hand. Second, Bundles is encouraged to learn meaningful, biologically-relevant mechanisms because of the fact that
similarly-structured compounds tend to have similar mechanisms. This provides an additional interpretability component to our predictor. We propose a new type of framework that melds existing machine learning methods with the traditional systems biology approach and, as a result, reduces the need for combination data altogether.
Chapter 2

Background

There is a range of computational methods used to model combinations of two drugs. Existing methods predict therapeutic activity [30, 5, 12], synergy [?, 23, 25], or adverse side effects of combinations [8, 32]. Methods for predicting activity and synergy are more closely aligned than methods for predicting side effects. This is because activity and synergy datasets are created from the same screens. They also tend to be much smaller than side effect datasets.

From a technical perspective, many existing methods fall into two broad categories: structure-based modeling and networks-based analysis. The first characterizes drugs using fixed molecular descriptors, such as Morgan fingerprints or Dragon descriptors [16]. Combinations are represented by the concatenation of the two drug descriptors. Then, traditional machine learning techniques, such as random forest and deep neural networks, are used to learn and predict properties. Many recent models also incorporate gene expression, microRNA, or proteome features to characterize the disease [23, 25, 30].

The transition from fixed to learned drug representations has led to many successes in the single-agent space [31, 29]. However, learned drug representations are not nearly as well studied when applied to combinations. So far, graph convolutional networks with co-attention have been used to predict adverse side effects on TWOSIDES, which comprises millions of labeled examples [8]. To our knowledge, learned drug representations have not been applied to activity or synergy prediction, for which
large datasets are less readily available.

The second category of existing methods uses a systems biology perspective [21]. These methods construct networks of drug-target and protein-protein interactions from literature and apply network science techniques [5] or graph convolutional networks [32] to this larger network. However, constructing the initial network is difficult, since interactions data may be unverified or incomplete.

A third nascent category includes methods that leverage single-agent data. To our knowledge, there exists one paper, in which they apply random forest classifiers to the means and deltas of the two combination components’ single-agent dose-response curves [9].

This thesis further explores the power of learned drug representations for predicting therapeutic activity of combinations. We carefully consider the trade-off between model size and the potential to overfit in order to develop architectures that are successful for activity prediction. In one of our models, we also integrate single-agent data, significantly improving the data efficiency of the method.
Chapter 3

Methods

In this chapter, we present two methods for modeling combinations. While combination therapies may refer to a mixture of any number of drugs, we focus on the two-drug cocktail. The first method, ComboFiLM, learns from combination data to predict any joint property (activity, synergism, side effects, etc.). It integrates information from both compounds in the combination through feature-wise linear modulations (FiLM) [22]. The second method, Bundles, learns solely from single-agent data to predict the therapeutic effect of a combination. During training, this model first builds an understanding of the biological latent mechanisms that drive therapeutic activity for single-agent drugs. For each compound in the combination, it predicts a behavioral profile that relates the drug to latent mechanisms. Then, it composes the two profiles to predict the combination’s joint effect.

For notation, a combination consists of compounds $A$ and $B$ and their respective concentrations $c_A$ and $c_B$. Our goal is to predict a joint property (e.g. therapeutic activity) of the combination.

3.1 ComboFiLM

We frame the combination problem as predicting behavior of compound $A$ in the context of $B$ or $B$ in the context of $A$. We first introduce two core ideas: learned drug representations and feature-wise linear modulation (FiLM). Then, we describe
how they are combined in our model.

3.1.1 Learned Drug Representations

Learned representations have been shown to be more effective than pre-computed, fixed descriptors when used in single-agent property predictors [31]. However, they have not been well studied for combination property predictors. In ComboFiLM, we use a message-passing neural network (MPNN) to construct learned representations for a given drug [31].

The representation of drug $A$ at concentration $c_A$, $\phi(A, c_A)$, is computed in two steps. First, given the molecular graph of a compound, the MPNN outputs two $d$-dimensional vectors, $w_0$ and $w_1$, where $d$ is the hidden size hyperparameter. Then, the compound’s concentration information is integrated as follows:

$$\phi(A, c_A) = w_0 + w_1 \log c_A$$  \hspace{1cm} (3.1)

We can think of $w_0$ as the baseline representation for a given drug. As the drug concentration changes, the drug may also behave differently, so the optimal representation for the drug also changes. This concentration-dependent modulation is represented by $w_1$. Since $w_0$ and $w_1$ are specific to each drug, the model can learn not only how the drugs differ from each other but also how a drug differs from itself at various concentrations.

Learning both $w_0$ and $w_1$ is important for modeling drugs at given concentrations, because drugs react to concentration changes in different ways. For example, alcohol is classified as a depressant but can behave as a stimulant at low concentrations. Other drugs, such as cocaine, are stimulants at any concentration.

As concentration level decreases ($\log c_A \to -\infty$), the drug representations do not necessarily converge onto one single “null” point. However, this discrepancy is never observed, because drugs are usually tested at a common, FDA-approved range of concentrations ($-14 \leq \log c_A \leq -1$ log-molar).
3.1.2 FiLM Layers

Feature-wise Linear Modulation (FiLM) is a popular technique that was initially developed for visual question-and-answering [22]. Generically, FiLM modulates a neural network by enabling a context to influence the forward propagation of the data through the network. In visual question-and-answering, the data corresponds to the image, whereas the context corresponds to the question. When we apply this technique to drug cocktails, the data and context become the two different compounds in the cocktail.

FiLM modulates the forward pass through a neural network with FiLM layers. Each FiLM layer applies an element-wise affine transformation to its input $x$, where the transformation is a function of the context $c$. A hypernetwork, called the FiLM generator, receives the context embedding $c$ and outputs $\gamma = g(c)$ and $\beta = h(c)$, where $g$ and $h$ are learned functions. The layer transforms input $x$ into output $\gamma \odot x + \beta$. This transformation is a generalization of other conditioning techniques, such as sigmoidal gating or concatenation conditioning.

3.1.3 Model Architecture

ComboFiLM architecture consists of the MPNN followed by classifier $f$, where $f$ alternates FiLM layers and linear layers (Fig. 3-1). First, the drugs are individually encoded into their representations $\phi(A, c_A)$ and $\phi(B, c_B)$ using the MPNN and Equation 3.1. Then, the representations are passed through classifier $f(\phi(A, c_A), \phi(B, c_B))$, which predicts the property of compound $A$ in the context of $B$. Because $f$ is asymmetrical, the model computes both $f(\phi(A, c_A), \phi(B, c_B))$ and $f(\phi(B, c_B), \phi(A, c_A))$ and outputs the mean. The model is trained using the following loss $\mathcal{L}$ and label $y$.

$$\mathcal{L}(\frac{1}{2}(f(\phi(A, c_A), \phi(B, c_B)) + f(\phi(B, c_B), \phi(A, c_A)), y)$$ (3.2)

Each FiLM layer consists of a generator that produces the affine transformation from the context. The parameters for the generator are not shared between FiLM
layers. To minimize the number of parameters in our model, the generators use linear functions, $g$ and $h$. For a forward pass conditioned on drug $B$, each FiLM layer receives input $x$ and outputs the following:

$$ReLU(g(\phi(B, c_B)) \odot x + h(\phi(B, c_B)))$$

(3.3)

The first FiLM layer in our model receives input $x = \phi(A, c_B)$.

After each FiLM layer is a fully-connected linear layer. Each linear layer receives and outputs a $d$-dimensional vector, where $d$ is the dimension of the drug representation. We use ReLU activations.

### 3.1.4 Training Procedure

We train the model end-to-end using early stopping for a maximum of 60 epochs with Adam (learning rate $10^{-3}$) and a batch size of 128. We also apply dropout before each FiLM layer, FiLM generator, and fully-connected linear layer. To address the large class imbalance, we subsample inactive combinations, such that every batch has an equal number of positive and negative examples.

We do a hyperparameter search with the test set held out using Bayesian optimization for 20 iterations. We optimize for area under the precision-recall curve.
(AUPRC) on the validation set. The hyperparameters are dropout probability, hidden size $d$, the depth of $f$ (number of pairs of FiLM and linear layers), and the number of message-passing iterations in the MPNN. Their ranges are shown in Table A.1.

3.2 Disease Bundles

Traditional machine learning approaches directly learn the relationship between combinations and therapeutic activity, which can be difficult given the limited size and compound diversity of combination data. In contrast, this model focuses on learning the biological processes relevant to each disease and predicts therapeutic activity from the drugs’ combined effect on these processes. From a machine learning perspective, these processes are equivalent to “latent mechanisms” that drive therapeutic activity. We refer to them as such for the remainder of the thesis. Since data limitations preclude us from directly learning the set of biological processes, we induce these latent mechanisms to represent biological concepts during training.

We first provide a high-level explanation of the model architecture. Then, we explain how the latent mechanisms are learned from single-agent data. Finally, we describe in detail how the model is trained and applied to combination therapies.

3.2.1 Model Framework

Our model, Bundles, consists of three components: drug representations, disease representations, and therapeutic activity prediction. Each component is guided by the following key insights from biology.

1. Chemical structure dictates how and why drugs affect various biological processes. As a result, drugs that have similar structures may affect the body through the same latent mechanisms.

2. Disease can be characterized by a set of abnormal biological processes or latent mechanisms.
3. Drugs treat a disease by addressing the underlying biological process. In a combination therapy, drugs may affect overlapping or complementary sets of processes. If they treat a complementary set, this may result in synergism [5].

Our model uses the same drug representations as the ComboFiLM model (Section 3.1.1). We encode the chemical structure and concentration information into a single $d$-dimensional vector using a message-passing neural network (MPNN) and Equation 3.1.

Next, we represent a disease with a set of $K$ vectors, where $K$ is a hyperparameter we choose (Section 3.2.2). We refer to this set of vectors as a “disease bundle” and an individual vector as a “bundle vector.” Our model may be trained on multiple diseases at the same time. In that case, each disease will have its own distinct set of $K$ bundle vectors.

Each $d$-dimensional bundle vector represents a latent mechanism specific to that disease. If a bundle vector is similar to a drug representation, that drug is predicted to treat the disease via the given latent mechanism. In our model, we measure similarity as cosine similarity. Applying cosine similarity between the drug $A$ embedding and each vector in the disease $D$ bundle, the model predicts a drug-disease profile $z_{A,D}$. Profile $z_{A,D}$ is a $K$-dimensional vector, where each dimension corresponds to the predicted impact the drug $A$ has on disease $D$ via the given latent mechanism.

Before discussing combinations, we explain how $z_{A,D}$ relates to single-agent activity of drug $A$ for disease $D$. Because a disease can be treated through any one of its latent mechanisms, the therapeutic activity of drug $A$ for disease $D$ is predicted to be:

$$\max_k z_{A,D} \quad (3.4)$$

While our model only has one MPNN, it can learn multiple disease bundles at once. However, during training or prediction, disease $D$ is always specified for the forward pass. Moving forward, we omit $D$ from the profile notation for simplicity.

Given single-agent profiles $z_A$ and $z_B$, Bundles uses Bliss independence to produce the combination profile $z_{A,B}$. Bliss independence is a well-known framework
that deduces the combination effect from two single-agent effects by assuming that the drugs act independently [3]. Our model produces profile $z_{A,B}$ by applying Bliss independence element-wise to the individual drug profiles, $z_A$ and $z_B$, where $\odot$ is element-wise multiplication:

$$z_{A,B} = z_A + z_B - z_A \odot z_B$$  \hspace{1cm} (3.5)

Each dimension of $z_{A,B}$ corresponds to the combination’s impact through the given latent mechanisms. Similar to Equation 3.4 for predicting single-agent activity, we predict combination activity to be:

$$\max_k z_{A,B}$$  \hspace{1cm} (3.6)

Figure 3-2: Visualization of the combination’s predicted impact for a latent mechanism, as a function of the single-agent mechanism impacts under the Bliss model of independence (left). Applying Bliss element-wise for all latent mechanisms causes two patterns of combination interactions to emerge (right). We explore this idea with a bundle size of 2. Each triangle represents the predicted mechanism impact. In interaction A, the combination has impact through one mechanism (1) but not the other (2). In interaction B, the combination has interaction through both mechanisms as a result of complementary single-agent impacts (3).
We visualize how Bliss would predict the joint impact of a mechanism, as a function of the two single-agent impacts (left, Fig. 3-2). There are three general scenarios that characterize predicted impact for a given mechanism:

1. Both single-agents are high impact, resulting in high predicted joint impact.

2. Both single-agents are low impact, resulting in low-to-medium predicted joint impact.

3. One single-agent is low impact while the other is high impact. This results in high predicted joint impact.

However, we note that applying Bliss always results in a higher predicted joint impact than the individual impacts. This generally biases the model to be more optimistic.

While the above discussion covers how Bliss affects joint impact per mechanism, we now focus on how it affects the overall combination profile and thus therapeutic activity prediction. Two types of intuitive combination interactions emerge from our application of Bliss. A combination can become active when two drugs build upon the same mechanism (top right, Fig. 3-2). Alternatively, the combination can develop multiple paths to activity if the individual drugs have complementary mechanisms (bottom right, Fig. 3-2). Because our primary goal is to model therapeutic activity, the model does not distinguish between these two interactions and simply predicts the combination’s activity to be \( \max_k z_{A,B} \). A potential future direction would be to investigate how these two scenarios translate to synergy predictions.

By assuming Bliss independence, our model can predict the activities of combination therapies in a zero-shot setting. Given drug and disease representations, we can compute single-agent profiles and merge them to get a combination profile and activity prediction. In order to learn drug and disease representations, we train on solely single-agent data across multiple different diseases. Using single-agent data confers several benefits. Compared to combination data, single-agent data is often more plentiful and covers a more diverse set of compounds. This enables our model to learn a variety of latent mechanisms for each disease, as well as effective drug representations.
3.2.2 Learning Latent Mechanisms

A disease bundle is a set of $K$ vectors, which each represent a latent mechanism that is relevant to the disease. Ideally, we would like to know this exact set of drug mechanisms and biological processes to use when training the model. However, mechanism-of-action data is incomplete and otherwise limited to a set of well-studied drugs.

To overcome this, we observe that similarly structured compounds have similar mechanisms. For example, antibiotics can generally be divided into different drug classes in which the molecular scaffold defines the mechanism of action (e.g. macrolides, fluoroquinolones, tetracyclines) [28]. During training, we divide active compounds into $K$ structural/functional groups and use each group to train a specific bundle vector. Because these groups are loose proxies for mechanism, the disease bundle is encouraged to learn different latent mechanisms that reflect biological concepts.

While our model may be simultaneously trained across many different diseases, the following clustering procedure is repeated for each disease, since latent mechanisms may vary per disease. From single-agent data, active compounds are clustered into $K$ groups by applying K-means to RDKit descriptors [14]. RDKit descriptors include features like log $P$ solubility or amide group counts. This divides the drugs based on global structural and functional trends, such that similarly structured compounds remain in the same cluster. Because compounds with similar structures often share mechanism of actions, these clusters approximate the underlying mechanisms.

Cluster assignments stay constant throughout training. Each bundle vector is only exposed to the drugs in its cluster. The model is trained using back-propagation, such that the drug embeddings of active compounds have high cosine similarity with their assigned bundle vector. Assuming that a cluster constitutes a drug class (defined as a pharmacological grouping of compounds with similar structures and mechanism), this training procedure encourages the bundle vector to look like a prototype drug for the drug class.
However, the cluster is only an imperfect approximation of a drug class. We argue that the model is robust to imperfect clusters, because the definition of the bundle vector can shift depending not only on the cluster assignments but also on the MPNN. Clusters provide implicit guidance for what the bundle vector ought to represent, whereas the drug representations produced by the MPNN directly influence bundle vectors during training. This provides flexibility in what is ultimately learned.

As a result, multiple bundle vectors may end up mapping to the same latent mechanism. If there are too many redundancies, either as a result of little data diversity or too many bundle vectors, then “bundle collapse” may occur. This risk can be monitored during single-agent training by calculating average pairwise cosine similarity between bundle vectors. Bundle collapse is explored further in Section 5.2.3.

$K$ should be chosen accordingly to prevent bundle collapse but also have enough capacity to model different latent mechanisms. Currently, Bundles uses the same $K$ for each disease for simplicity, but it can be easily extended to have different sized bundles per disease. In this work, we select $K = 200$ based on what we observed with bundle collapse on our specific dataset (Section 5.2.3).

### 3.2.3 Training Procedure

During training, the model learns drug representations and disease bundles from single-agent data. Our model is trained on data across multiple diseases. This structure is similar to the recommender problem between drug and disease, except the disease is represented by a set of vectors instead of a single embedding. Additionally, these vectors are latent mechanisms that are induced to represent biological concepts of the disease. The drug representations are computed from two components: a MPNN that encodes structure and Equation 3.1 that integrates concentration information. The drug representation is the same one used in ComboFiLM (Section 3.1.1).

A recommender model typically predicts a numerical score for the user and item. Here, our model predicts a $K$-dimensional disease-drug profile, where each dimension represents the impact a drug has via a given latent mechanism. Our model is trained end-to-end from scratch using back-propagation on a series of (drug, concentration,
activity, disease) tuples. For active tuples, our model looks up the disease-specific cluster assignment for the drug. It outputs the drug impact for its assigned latent mechanism, which is trained against a positive label. For inactive tuples, our model learns that the drug impact for all latent mechanisms in the disease bundle should be zero across the profile (Fig. 3-3). This is because inactive compounds should not be act through any latent mechanism of the disease.

We use an Adam optimizer with a learning rate of $10^{-3}$. We train for 30 epochs with a batch size of 256. Each batch is class-balanced, such that it contains an equal number of active and inactive tuples. As a result, inactive tuples are subsampled (globally, not per disease). In this work, we use a bundle size of 200 (Section 5.2.3). We also select a hidden size of 75 and use three iterations of message passing, based on parameters that worked well for the MPNN in single-agent applications.

Overall, our workflow is as follows. First, we train our model on single-agent data. It learns the latent mechanisms underlying effective single-agent treatments across different diseases. Then, we apply Bundles zero-shot to combination therapies. For a given disease, our model predicts a $K$-dimensional single-agent drug profile
Figure 3-4: Illustration of how disease bundles are used to predict combination activity for a given disease. Single-agent profiles $z_A$ and $z_B$ are derived for each compound independently by comparing a given drug representation to bundle vectors. $f(z_A, z_B)$ produces combination profile $z_{A,B}$ by applying Bliss independence to each dimension. The triangles represent impact per latent mechanism. The max value in $z_{A,B}$ is the predicted activity for the cocktail.

that corresponds to the similarity of the drug representation to the various bundle vectors. A single-agent drug is said to act via a given latent mechanism if their representations are similar. Given the profiles of drugs $A$ and $B$, $z_A$ and $z_B$, we apply Bliss independence to each dimension to produce the combination profile (Eq. 3.5). We can then screen for combination profiles that have a high impact in one or more latent mechanisms. The final predicted activity for the combination is $\max_k z_{A,B}$ (Fig. 3-4).

Although we focus on combinations of two drugs, this model can be extended easily to cocktails of $N$ drugs, provided there is a correct inductive bias for how to deduce the joint activity vector from the individual compound activity vectors.
Chapter 4

Experimental Procedure

We evaluate the predictive ability of our models to identify therapeutically active combination therapies. The problem is posed as a classification task given a combination of two drugs and their concentrations.

4.1 Data

We test our models on the NCI-ALMANAC dataset, a high-throughput screen of two-drug cocktails on cancer cell lines [11]. It is the one of the largest publicly available combination datasets. A data point is a tuple consisting of (drug 1, drug 2, drug 1 concentration, drug 2 concentration, percent growth of the cancer, ComboScore, and cell line). ComboScore is a proxy for synergy in this dataset. It is calculated as the difference between expected growth and observed percent growth.

In total, the dataset contains hundreds of thousands of tuples and covers 60 cell lines across 9 cancers. Each assay, performed on a cell line, screens roughly 5,000 unique combination therapies from a library of 105 SMILES strings. Each unique pair of drugs is tested on a 3-by-3 grid of concentrations. Table 4.1 contains a basic description of the dataset.

In our classification task, a combination is labeled as therapeutically active if percent growth is less than 0. Negative percent growth is desirable, because it means that the cancer growth is inhibited.
<table>
<thead>
<tr>
<th>Cancer</th>
<th>Cell Line</th>
<th>Data Size</th>
<th>% Hits</th>
<th>Unique Pairs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Breast</td>
<td>MCF7</td>
<td>46,953</td>
<td>1.8%</td>
<td>5,001</td>
</tr>
<tr>
<td></td>
<td>ATCC</td>
<td>46,848</td>
<td>3.9%</td>
<td>4,991</td>
</tr>
<tr>
<td>CNS</td>
<td>SF-268</td>
<td>46,908</td>
<td>5.5%</td>
<td>4,994</td>
</tr>
<tr>
<td></td>
<td>SNB-19</td>
<td>46,836</td>
<td>4.5%</td>
<td>4,986</td>
</tr>
<tr>
<td>Colon</td>
<td>HCT-116</td>
<td>46,893</td>
<td>2.6%</td>
<td>4,943</td>
</tr>
<tr>
<td></td>
<td>SW-620</td>
<td>47,106</td>
<td>3.1%</td>
<td>5,016</td>
</tr>
<tr>
<td>Leukemia</td>
<td>RPMI-8226</td>
<td>44,850</td>
<td>21.1%</td>
<td>4,771</td>
</tr>
<tr>
<td>Melanoma</td>
<td>SK-MEL-28</td>
<td>46,854</td>
<td>4.6%</td>
<td>4,989</td>
</tr>
<tr>
<td></td>
<td>MALME-3M</td>
<td>46,035</td>
<td>6.4%</td>
<td>4,933</td>
</tr>
<tr>
<td>Lung</td>
<td>NCI-H23</td>
<td>46,845</td>
<td>11.4%</td>
<td>4,987</td>
</tr>
<tr>
<td></td>
<td>NCI-H322M</td>
<td>46,257</td>
<td>4.3%</td>
<td>4,926</td>
</tr>
<tr>
<td>Ovarian</td>
<td>OVCAR-5</td>
<td>46,920</td>
<td>4.4%</td>
<td>4,997</td>
</tr>
<tr>
<td></td>
<td>OVCAR-4</td>
<td>46,218</td>
<td>5.3%</td>
<td>4,920</td>
</tr>
<tr>
<td>Prostate</td>
<td>PC-3</td>
<td>46,488</td>
<td>3.9%</td>
<td>4,958</td>
</tr>
<tr>
<td>Renal</td>
<td>ACHN</td>
<td>47,106</td>
<td>9.6%</td>
<td>5,003</td>
</tr>
</tbody>
</table>

Table 4.1: Summary statistics per cell line in the NCI-ALMANAC dataset. We only include the cell lines that we use in evaluation. A hit is a therapeutically active combination. Unique pairs refers to the number of unique combinations of SMILES.

Our experiments prioritize evaluating predictive ability for percent growth (i.e. activity) over ComboScore for several reasons. A combination with a positive ComboScore (i.e. synergistic) is not necessarily therapeutically active. Out of the combinations with positive ComboScores, only 13% inhibit cancer growth. Furthermore, a previous study demonstrated that variability of percent growth data introduces noise into ComboScore and impairs modeling [25]. The variability also impacts our task, but not to the same extent. When pre-processing the data, we use the mean percent growth of duplicate tuples, which reduces experimental error. On the other hand, for some ComboScores, expected growth is calculated by multiplying percent growth data. This compounds experimental error.

Synergy is relevant for our task, because we need to identify the effective combinations that are active in spite of the inactivity of their individual components. To address this, we divide the data into tiers that represent how likely a combination is to be successful and report model performance on each tier. Based on single-agent data, a combination is assigned to either Tier 0, 1, or 2. Tier 2 combinations consist
Table 4.2: Tier-specific summary statistics per cell line in the NCI-ALMANAC dataset. Cell lines above the bold line are handpicked for evaluating Bundles.

<table>
<thead>
<tr>
<th>Cancer</th>
<th>Cell Line</th>
<th>Tier 0</th>
<th>Tier 1</th>
<th>Tier 2</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Size</td>
<td>Hits</td>
<td>Size</td>
<td>Hits</td>
</tr>
<tr>
<td>Breast</td>
<td>ATCC</td>
<td>10,287</td>
<td>2.6%</td>
<td>23,088</td>
</tr>
<tr>
<td>CNS</td>
<td>SNB-19</td>
<td>12,831</td>
<td>2.3%</td>
<td>23,562</td>
</tr>
<tr>
<td>Colon</td>
<td>SW-620</td>
<td>12,078</td>
<td>1.0%</td>
<td>23,733</td>
</tr>
<tr>
<td>Melanoma</td>
<td>MALME-3M</td>
<td>9,027</td>
<td>1.7%</td>
<td>22,542</td>
</tr>
<tr>
<td>Leukemia</td>
<td>RPMI-8226</td>
<td>8,811</td>
<td>8.3%</td>
<td>22,197</td>
</tr>
<tr>
<td>Lung</td>
<td>NCI-H322M</td>
<td>12,879</td>
<td>2.1%</td>
<td>23,247</td>
</tr>
<tr>
<td>Ovarian</td>
<td>OVCAR-4</td>
<td>10,062</td>
<td>4.4%</td>
<td>23,217</td>
</tr>
<tr>
<td>Prostate</td>
<td>PC-3</td>
<td>13,419</td>
<td>1.6%</td>
<td>22,803</td>
</tr>
<tr>
<td>Renal</td>
<td>ACHN</td>
<td>8,979</td>
<td>3.0%</td>
<td>22,887</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Cancer</th>
<th>Cell Line</th>
<th>Tier 0</th>
<th>Tier 1</th>
<th>Tier 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Breast</td>
<td>MCF7</td>
<td>9,699</td>
<td>0.1%</td>
<td>22,422</td>
</tr>
<tr>
<td>CNS</td>
<td>SF-268</td>
<td>8,577</td>
<td>1.4%</td>
<td>22,740</td>
</tr>
<tr>
<td>Colon</td>
<td>HCT-116</td>
<td>8,613</td>
<td>0.6%</td>
<td>22,764</td>
</tr>
<tr>
<td>Melanoma</td>
<td>SK-MEL-28</td>
<td>9,342</td>
<td>0.2%</td>
<td>22,9865</td>
</tr>
<tr>
<td>Lung</td>
<td>NCI-H23</td>
<td>6,705</td>
<td>1.7%</td>
<td>21,882</td>
</tr>
<tr>
<td>Ovarian</td>
<td>OVCAR-5</td>
<td>9,009</td>
<td>1.6%</td>
<td>23,319</td>
</tr>
</tbody>
</table>

Table 4.2 displays a tier-based breakdown of the data.

Single-Agent Data

In addition to combination data, we also use single-agent data for training Bundles and assigning combinations to tiers. NCI-60 is the single-agent predecessor to NCI-ALMANAC [24]. This dataset covers a superset of the cell lines and compounds used in the combination screens. Each assay contains hundreds of thousands of (drug, concentration, percent growth) tuples with around 40,000 unique SMILES.
Pre-processing

Both NCI datasets are labeled with NSC identifiers. We mapped NSC identifiers to SMILES using the PubChem ID Exchange [13]. This process recovered 103 of 105 compounds in the combination data and 48,665 compounds in the single-agent data. Duplicates were handled by taking the mean percent growth.

4.2 Evaluation Framework

We specifically want our models to be able to generalize to new combinations that involve compounds not seen in the training set (i.e. out-of-library compounds). To that end, we use cross-validation with a Leave-Out-Compound (LOCO) split. In LOCO, the unique SMILES strings in data are first randomly allocated to the different splits. Then, the combinations of the associated SMILES follow thereafter. If a combination is composed of two SMILES that belong to different splits, then preference is given to test, validation, and training in that order (Fig. 4-1). We use an 80-10-10 partition for allocating SMILES to training, validation, and test sets.

Many existing methods use random split, which is quite different from LOCO. In random split, the combinations themselves are randomly allocated to train and test. Random split would be not be representative of performance on out-of-library combinations, since there may be information leak on how a compound would perform in combination with others. We observe that LOCO split is not only more difficult but also more variable than random split. Performance can vary substantially from fold-to-fold, depending on how the SMILES are allocated. So we would recommend using cross-validation when splitting on compounds to get an accurate representation of performance.

We assess performance using both area under the receiver operating characteristic curve (AUROC) and area under the precision-recall curve (AUPRC). AUPRC is the primary metric used for evaluation, since it accounts for the class imbalance in the dataset. AUPRC is calculated using the average precision method, which provides a more accurate estimate than the typical trapezoidal rule.
Figure 4-1: Visualization of leave-out-compound split (left) and random split (right). Combination data can be viewed as a drug-drug matrix of labels.

### 4.2.1 Evaluating ComboFiLM

To save on computation time, we selected a random cell line from each cancer type. Each cell line is treated as a separate dataset, so each model is trained from scratch on the relevant subset of combination data. The 9 selected cell lines are SK-MEL-28, SF-268, NCI-H23, HCT-116, MCF7, OVCAR-5, ACHN, RPMI-8226, PC-3.

We evaluate using 3-fold cross validation. This balances the required computation load with the need to account for variability between the splits. For each fold, we first divide the dataset into 80% training, 10% validation, and 10% test using a randomly-seeded LOCO split. Then, we tune model hyperparameters for 20 iterations using Bayesian optimization. During tuning, the test set is held out. The best hyperparameters are selected by optimizing for validation AUPRC. Finally, we use the optimized hyperparameters to train the last model with the original train, validation, and test sets. We report performance on the test set from this final model.

#### Baselines

We compare ComboFiLM against the following baselines to demonstrate the power of learned representations and FiLM layers respectively.
1. **Morgan** is a baseline that uses fixed drug representations. Morgan Count descriptors are computed for the two drugs. The descriptors are concatenated, along with their log concentrations, and fed through fully-connected layers. This is a simplification of an existing method that uses cell line features [30].

2. **MPNN** uses the same architecture as ComboFiLM, except without the FiLM layers. One MPNN encodes the structure of both drugs in a given combination. Their representations are concatenated along with the log concentrations and fed through fully-connected layers. We use this baseline to demonstrate the value of FiLM layers.

Both architectures are inherently asymmetric. To address this, the baselines models output the mean prediction that result from the concatenation of $A, B$ and concatenation of $B, A$.

The baselines are trained using Adam with a learning rate of $10^{-3}$ and a batch size of 256 that is balanced similarly to ComboFiLM. We also used Bayesian optimization to tune model hyperparameters. For Morgan, the hyperparameters are dropout probability, number of the fully-connected layers, and hidden size of the layers. For Morgan, the hyperparameter set also includes the number of message-passing iterations. The ranges of the hyperparameters are adjusted to match the same possible model sizes as ComboFiLM.

### 4.2.2 Evaluating Disease Bundles

We evaluate overall and tiered performance of *Bundles* using 5-fold cross validation on LOCO split. Although the model is trained simultaneously on single-agent data across multiple different diseases (i.e. cell lines), we test and report performance on combination data separately for each cell line. Tiered AUC is calculated using only the subset of data that belongs to the given tier.

The cell lines used to evaluate *Bundles* are different from the subset used for ComboFiLM. For each cancer, we selected the cell line with the largest Tier 0 hit.
percentage. This criteria ensures there are enough active compounds to get an accurate estimate of Tier 0 AUPRC. If we were to choose cell lines randomly, as in the ComboFiLM procedure, some LOCO splits would have no active combinations in Tier 0, resulting in an undefined AUPRC. The 9 assays used to evaluate Bundles are listed above the bold line in Table 4.2.

We evaluate Bundles in two different modes. In the zero-shot setting, the model is trained solely on single-agent data and tested on combination data. In the data-scarce setting, the model is allowed to finetune on combination data. We note that both settings share the same (pre-)training procedure on single-agent data. The model generally performs well on the NCI-60 single-agent data, reaching 0.62-0.7 AUPRC and > 0.9 AUROC on each cell line.

**Zero-Shot**

Bundles is trained on single-agent data that includes the different cell lines. We select the model that performs the best on single-agent validation split. During testing, we use LOCO split to divide the combination data into train, validation, and test. AUPRC and AUROC are calculated on test data. Although the zero-shot setting does not require train and validation, this setup allows us to make equal comparisons to the baseline models described below.

**Data-Scarce**

In the data-scarce setting, Bundles is finetuned on combination data. The same training procedure from the zero-shot setting is used for single-agent training. Then, using this pre-trained model as an initialization point, the model is trained on combination data for a given cell line. By training a new model for each cell line, the encoder can be finetuned on the specific assay.

When training on combination data, we use a similar procedure as described in Section 3.2.3. Inactive combinations back-propagate through the entire bundle, because they should have low impact on all latent mechanisms. Meanwhile, we cannot discern which latent mechanisms ought to be responsible for a combination. Active
combinations back-propagate only through the predicted output, the bundle vector with the maximum impact.

Using 5-fold cross validation, we split the combination data into train, validation, and test sets using LOCO split. In order to assess how robust the model is to small datasets, we expose the model to either 10%, 25%, 50%, 75%, or 100% of the SMILES in the combination data. For the varying levels of exposure, the training and validation set involves 9, 22, 45, 70, and 92 unique SMILES respectively. No matter the exposure level, the test set always contains the combinations of 10-11 unique SMILES. All combinations that are associated with these SMILES are included in evaluation. The sizes of the training, validation, and test sets are shown in Table A.2.

**Baselines**

We compare Bundles against the following baselines.

1. **Oracle** is a model that is trained on combination data and directly learns the relationship between combination and its label. It provides an idea for reasonable performance. The model has the same architecture as the MPNN baseline used in ComboFiLM evaluation. To match the model size of Bundles, we set its hidden size to 75 and use two fully-connected layers. We train a separate model for each cell line over 30 epochs using an Adam optimizer with $10^{-3}$ learning rate.

2. **Single** demonstrates the value of representing diseases with a set of vectors. It has the same architecture and training procedure as Bundles, except it uses a bundle size of 1, instead of 200. We note that this setup reduces single-agent training to the recommender problem between compounds and diseases.

3. **Random** illustrates the importance of RDKit clustering. In this baseline, the groups for each bundle vector are formed around $K$ randomly selected cluster centers. Its architecture and training procedure are otherwise the same as Bundles.
Interpretability

Ideally, the disease bundle should represent a diverse set of biological processes. We run additional experiments to further probe this claim. To first assess diversity, we test how closely related the bundle vectors are for any given disease. If the bundle vectors are near in high-dimensional space, then they are more likely to represent the same latent mechanism. For each disease, we characterize nearness by analyzing the minimum volume enclosing ellipsoid that contains the bundle vectors. The ellipsoid is found by solving the equivalent linear optimization problem [20].

For each disease, we also validate that learned latent mechanisms correspond to real biological mechanisms. This evaluation is performed on single-agent compounds. For a pair of compounds, we compute their inferred similarity, based on latent mechanisms, and their bio-similarity, based on experimentally validated drug mechanism data. If the inferred similarity and bio-similarity measures correlate, then this implies that latent mechanisms have biological significance.

To calculate inferred similarity, the model first predicts the drug’s $K$-dimensional profile at its maximum observed concentration. Each dimension of this profile corresponds to the probability the drug acts through the given latent mechanism. Let the profiles for the two drugs be $z_A$ and $z_B$. The inferred similarity is the one minus the L1 norm of the difference:

$$1 - \|z_A - z_B\|_1$$

Given experimental mechanism data, the bio-similarity between two compounds is proportional to the number of mechanisms they share. Let $m_A$ and $m_B$ be bit vectors where 1 denotes that the drug acts through the given mechanism. Bio-similarity is calculated as follows, where & is the bitwise operator AND:

$$\frac{m_A \& m_B}{\min(\|m_A\|_1, \|m_B\|_1)}$$

The numerator counts how many mechanisms they share. The denominator is a normalization factor that accounts for the fact that drugs may be active in a varying
number of mechanisms. This especially needed, because our data is incomplete and may not contain the full set of mechanisms for any given drug.

Our experimental data consists of drug-process associations that are recovered from drug-target bioassays in CHEMBL [17]. Because targets are sparse, they are aggregated together under more general gene ontology (GO) terms. GO terms are nodes in a crowd-sourced, hand-curated hierarchy of biological concepts [7]. They represent gene functions or biological processes. The hierarchy can be thought of as a directed acyclic graph, where the leaves are the bioassay targets.

We retrieve all human bioassays related to the SMILES in our combination dataset and filter for only the bioassays with labeled activity in the comments section. This filter is necessary due to the lack of standardization in CHEMBL data. This process recovered target data for 26 out of the 103 SMILES strings.

For each target associated with a drug, the target is propagated up the GO hierarchy, such that all its GO term ancestors are also linked to the drug (Fig. 4-2). We additionally filter for mid-level hierarchy GO terms. We define mid-level GO terms as those that are linked to 5-11 SMILES. This range is picked based on our data (Fig. B-1). We remove sparse low-level GO terms and generic high-level GO terms to which all compounds are indiscriminately linked. We also include SMILES for which we have enough target data (i.e. at least 25 GO terms). At the end, our mechanism dataset contains 16 SMILES over 202 GO terms. The number of GO terms matches our model’s bundle size of $K = 200$. 

Figure 4-2: A toy hierarchy of GO terms (rectangles), as related to targets (circles) and compounds. The blue box contains the links collected from CHEMBL bioassays. After propagating targets through the GO hierarchy, the right compound’s mechanisms consist of the bold rectangles. The left compound’s mechanisms are the bold and dotted rectangles.
Chapter 5

Results

5.1 ComboFiLM

While AUPRC and AUROC are reported per disease, we also aggregate them to assess overall performance when comparing models. Figure 5-1 shows that ComboFiLM outperforms both baseline models in both AUPRC and AUROC. For each disease, ComboFiLM achieves up to a 350% gain in AUPRC over Morgan and 100% gain over MPNN. ComboFiLM consistently surpasses MPNN by at least one standard deviation (Table 5.1). The one exception is cell line RPMI-8226, on which they match performance.

We believe that these performance gains stem from ComboFiLM’s greater model capacity. In our model, information from the second compound can affect computation even in later layers of the classifier. This is in contrast to the baselines and other existing methods, where compound information is integrated only once through concatenation at the start of the fully-connected layers. A significant proportion of the performance gain can also be attributed to the use of learned drug representations. MPNN outperforms Morgan by a wide margin on all cell lines except for RPMI-8226.

We analyze how model performance is affected by the class imbalance of the dataset. Figure 5-2 shows that ComboFiLM is relatively robust to class imbalance, despite its greater model capacity. ComboFiLM consistently achieves AUPRC above 0.4, while the other baselines tend to fall below that threshold. We hypothesize that
Figure 5-1: Boxplot of model performances on different cell lines. Performance is measured in AUPRC (left) and AUROC (right).

<table>
<thead>
<tr>
<th>Cell Line</th>
<th>Morgan</th>
<th>MPNN</th>
<th>ComboFiLM</th>
</tr>
</thead>
<tbody>
<tr>
<td>SK-MEL-28</td>
<td>0.303 ± 0.078</td>
<td>0.413 ± 0.108</td>
<td>0.560 ± 0.083</td>
</tr>
<tr>
<td>SF-268</td>
<td>0.275 ± 0.104</td>
<td>0.384 ± 0.050</td>
<td>0.440 ± 0.007</td>
</tr>
<tr>
<td>NCI-H23</td>
<td>0.490 ± 0.064</td>
<td>0.577 ± 0.129</td>
<td>0.690 ± 0.095</td>
</tr>
<tr>
<td>HCT-116</td>
<td>0.129 ± 0.044</td>
<td>0.218 ± 0.088</td>
<td>0.281 ± 0.068</td>
</tr>
<tr>
<td>MCF7</td>
<td>0.100 ± 0.015</td>
<td>0.226 ± 0.117</td>
<td>0.453 ± 0.027</td>
</tr>
<tr>
<td>OVCAR-5</td>
<td>0.203 ± 0.076</td>
<td>0.394 ± 0.002</td>
<td>0.444 ± 0.086</td>
</tr>
<tr>
<td>ACHN</td>
<td>0.330 ± 0.046</td>
<td>0.427 ± 0.041</td>
<td>0.573 ± 0.060</td>
</tr>
<tr>
<td>RPMI-8226</td>
<td>0.675 ± 0.073</td>
<td>0.615 ± 0.132</td>
<td>0.573 ± 0.161</td>
</tr>
<tr>
<td>PC-3</td>
<td>0.165 ± 0.011</td>
<td>0.292 ± 0.128</td>
<td>0.481 ± 0.107</td>
</tr>
</tbody>
</table>

Table 5.1: Performance of ComboFiLM compared to baselines. Mean AUPRC and standard deviation on 3-fold cross validation are reported for each cell line.
the FiLM layers may force the drug encoder to be more robust. Not only are the drug embeddings used as input to the fully-connected layers, they must also be useful to the FiLM generators at varying depths. Because the embeddings are used throughout the net and for multiple purposes, the MPNN may be less prone to overfitting and is comparatively better than the regular MPNN in highly imbalanced settings.

5.2 Disease Bundles

In this section, we present the performance of Bundles in a zero-shot and data-scarce setting. Then, we perform additional experiments to understand what the latent mechanisms encode.

5.2.1 Zero-Shot Setting

Figure 5-3 displays aggregate model performance across cell lines relative to Oracle’s AUPRC. Oracle provides an idea for how a reasonable model would perform when given access to combination data. Because performance can vary significantly
depending on cell line, we use Oracle’s performance as an anchoring point for our comparisons. As expected, the zero-shot models either match or lag behind the Oracle.

Despite training solely on single-agent data, Bundles performs roughly the same as the Oracle. Single and Random both perform worse with a median $-0.1$ AUPRC. This demonstrates the importance of using a multiple-vector disease representation and RDKit clustering to guide training. Both are critical components that work in tandem to help the model learn the different latent mechanisms of diseases. Because Bundles roughly matches performance of the Oracle, RDKit clustering seems sufficient for the model to recover these meaningful relationships between drug, mechanism, and disease. At the same time, we hypothesize that Bundles would perform even better if we could train latent mechanisms using known mechanism clusters instead of RDKit-based proxies.

Bundles outperforms Single and Random across the different tiers (Fig. 5-4). Compared to the Oracle, Bundles matches the AUPRC in Tier 0 and Tier 1 but does worse
Figure 5-4: Boxplot of overall and tiered performances on different cell lines. Performance is measured in AUPRC. Tier performance is calculated using only the combinations in the test set that corresponds to the given tier.
in Tier 2. There are two reasons that account for the lag in Tier 2. First, we would expect Oracle to do better on Tier 2, since Tier 2 has the least class imbalance. Second, Bundles is optimistically biased on Tier 2 combinations. Due to Bliss independence, the joint impact of a particular mechanism is always at least the maximum of the two individual impact. This overlooks antagonism, likely leading to a comparatively higher incidence of Tier 2 false positives.

On the other hand, Bundles is pessimistically biased on Tier 0 combinations. This stems from the fact that the model is trained solely on single-agent data. A combination belongs to Tier 0 if both of its compounds have no observed activity in the single-agent data. The model is trained to show low impact across all latent mechanisms. So Bundles is primed to predict Tier 0 combinations as inactive.

However, Bundles is able to overcome its pessimistic bias and match performance on Tier 0. This is due to the combined power of Bliss independence and latent mechanisms. For a given latent mechanism, Bliss independence would predict a high joint impact given two individual medium impacts. However, in order for a high joint impact to be produced, the profiles of the single-agent have to align, such that the medium impacts occur for the same latent mechanisms. This highlights the need for our bundle to encode latent mechanisms that are meaningful.

Another factor that enables good Tier 0 performance is the nature of single-agent training. Although our loss function pushes the predicted impacts towards 0, the model in practice only learns impacts that are sufficiently low enough (i.e. lower than the active impacts). As a result, inactive compounds can still have muted level of impact in some latent mechanisms but not others. This variability is leveraged by Bliss independence and latent mechanisms as described above.

We further assert that the success of Bundles is not a circumstantial result of imperfect modeling of single-agent data. Random and Single similarly have imperfect understanding of single-agent data. However, their combination performances are far worse than that of Bundles. This demonstrates the importance of having latent mechanisms that are biologically significant and thus allow the single-agent profiles to combine in meaningful ways for combination predictions.
The dynamics of tiered performance highlight a key advantage: Bundles can be used to identify active Tier 0 combinations at little-to-no marginal cost. Integrating the model into the combination development process would have low cost and high potential upside. Currently, the Tier 0 combination space is currently vastly unexplored. From single-agent data, labs allocate resources to combinations that they think are likely to succeed, which biases the tested combinations towards Tier 1 and Tier 2. This means that active combinations in Tier 0 are often overlooked and rarely discovered. Bundles sifts through the vast combinatorial Tier 0 space to identify the select few by leveraging single-agent data alone. As a result, it can produce a cost-effective set of candidate combination therapies that would not have been discovered otherwise.

5.2.2 Data-Scarce Setting

In this setting, Bundles is first trained on single-agent data and separately finetuned on combination data for each disease. We benchmark against the Oracle and Single. Because the Oracle is no longer an oracle in this experimental setting, we refer to it as MPNN in this section. Single is also a crucial baseline, because it is pretrained on single-agent data while MPNN is not. To assess robustness to dataset size, the models are exposed to a percentage of available training data, ranging from 10% to 100%.

By finetuning combination data, Bundles reaches a median AUPRC of 0.6 across the different diseases (Table A.4). This surpasses the performance of both baselines. At every exposure level, Bundles and Single both significantly outperform MPNN (Fig. 5-5) in AUPRC. The most significant gaps are seen on smaller training sets. This suggests that pre-training on single-agent datasets can help methods become more robust to class imbalance or small datasets. Because single-agent datasets are typically larger and more diverse than combination assays, it is reasonable that combination models would benefit significantly from this data source. No matter the model architecture, we would recommend pretraining on single-agent data.

Bundles also outperforms Single across all exposure levels. This shows that the
Figure 5-5: AUPRC (top) and AUROC (bottom) performances varying with the percentage of dataset exposed to the model. Points correspond to the median of AUC for the different diseases. Error bars correspond to the upper and lower quartiles.
set-of-vectors is a more effective representation for diseases. However, we note that the performance gap slightly narrows as more data is used— from 0.17 at zero-shot to 0.09 when finetuning on 100% of combination data. One possible explanation is that the training procedure for combinations does not effectively utilize the latent mechanisms. Currently, active combinations only back-propagate through the bundle vector with the highest joint impact, which may not correspond to the real latent mechanism driving combination activity. The lack of specificity contrasts the single-agent training procedure, in which bundle vectors develop structure through their specific, designated group of compounds. This reinforces the fact that bundle representations are effective only when the training provides enough guidance to learn latent mechanisms.

We observe different behaviors with AUPRC and AUROC. While AUPRC increases substantially with more training data, AUROC remains nearly identical for both the pretrained models, Bundles and Single (Fig. 5-5). Judging from AUROC, the model seems to gain reasonable understanding from single-agent data alone. Then, the combination data helps re-calibrate the model to be more sensitive to active combinations. We hypothesize that training on combinations particularly lowers the incidence of false positives in Tier 1 and 2. After finetuning, the increases seen in Tier 1 and 2 performances are two-fold compared to the increase in Tier 0 AUPRC (Fig. B-2). For Tier 1 and 2, the model is biased towards optimism because of the nature of Bliss independence. Leveraging combination data to back-propagate through all bundle vectors, the model can effectively learn from the large number of inactive combinations and reduce the number of false positives.

5.2.3 Interpreting Bundle Vectors

We delve deeper to understand what the latent mechanisms are encoding. Our experiments aim to understand the diversity and biological significance of the disease bundle.
Bundle Collapse

The opposite of bundle diversity is bundle collapse. “Bundle collapse” occurs when the disease bundle learns overly redundant latent mechanisms. This may occur if the single-agent data does not cover a diverse enough set of compounds. Another possible scenario is if bundle size $K$ is too small. If $K$ is not sufficiently large, the RDKit clusters may overlap too much and prevent the model from learning the distinct mechanisms. In the extreme case $K = 1$, there is one large RDKit cluster whose shared property is therapeutic activity. The model ultimately learns the direct relationship between combination and activity instead of the different reasons for activity.

To demonstrate when bundle collapse can occur, we train two different models with $K = \{50, 200\}$ and analyze their bundle diversities. For each model and a given disease, we calculate the minimum volume enclosing ellipsoid (MVEE), such that the disease bundle is contained within the ellipsoid. A MVEE with larger radii (even along one dimension) is more desirable than a MVEE with smaller radii. Larger radii would mean that the latent mechanisms are represented by different regions of the high-dimensional space and thus are different from one another.

For the model with $K = 50$, the radii of this ellipsoid are incredibly small. Across all diseases, the maximum observed radius is on the order of $10^{-1}$. On the other hand, the MVEE radii of the $K = 200$ model shows greater range. The MVEE for each disease consistently has multiple radii around length 10 (Fig. 5-6). Compared to its $K = 200$ counterpart, a $K = 50$ disease bundle uses a much smaller space and thus likely learns many redundant latent mechanisms. This is an example of bundle collapse. We hypothesize that this is due to the fact that the bundle size is too small, causing the RDKit clusters to be too similar to guide latent mechanism training.

We rule out the idea that the $K = 50$ model operates on a different scale. Both models, regardless of bundle size, use roughly the same volume of high-dimensional space across all diseases. This is verified by computing the MVEE that encloses the bundle vectors from all diseases. In fact, both of the models’ global MVEE’s have
Figure 5-6: Distribution of radii for the MVEE’s. Larger radii (trending to the right) are more desirable. The ellipsoids are calculated for the PC-3 disease bundle from two different models ($K = 50$ and $K = 200$). Radii for other disease bundles follow similar distributions.

A distribution of radii that looks similar to the $K = 200$ distribution. This implies that our model, with a bundle size of 200, learns a diverse set of mechanisms for each disease. On the other hand, the model with bundle size 50 learns to carve out a small space within its global MVEE for each disease.

**Biological Significance**

Using the collected drug-mechanism dataset and predictions from our trained single-agent model, we calculate the bio-similarity and inferred similarity measures between every pair of the 16 available SMILES. Then, for each disease, we compute Spearman’s rank-order correlation between the similarities.

For *Bundles*, the correlation varies from 0.22-0.23. For Random, the correlation varies from 0.03-0.06. The disparity between the two demonstrates the important role of RDKit clusters in guiding and training latent mechanisms.

While *Bundles* has statistically significant correlations with p-values < 0.01, its absolute correlation may be low due to several reasons. First, we note that the drug-mechanism dataset is agnostic to disease. The GO terms in the dataset involve generic
cancer processes, such as DNA replication or other various signaling kinases. On the other hand, the model learns latent mechanisms that are curated to each specific disease. This adds different biases to the bio- and inferred notions of similarity. Second, due to data limitations, these similarities are calculated for a relatively small and biased subset of available SMILES. The SMILES in the mechanism dataset are the most well-studied compounds. As a group, they are likely less mechanistically diverse than the full set of compounds in NCI combination data.

While the limitations of the drug-mechanism data are not ideal, our model has comparatively more biological significance than the Random baseline. This experiment demonstrates that RDKit clustering is able to guide the model to learn latent mechanisms towards meaningful biological concepts. The correlations cannot be strictly attributed to chance.

This experiment could be improved by acquiring better or more complete drug-mechanism data. However, given this data, we could also use it to train the model directly instead of using it to validate. As long as there is a need for validating biological significance ex post facto, we can only reconstruct a partial view into what the latent mechanisms mean to the model. Future work includes better validating latent mechanisms and proposing a more optimal clustering to guide bundle vectors.
Chapter 6

Conclusion

In this work, we propose two different methods for identifying active combination therapies. ComboFiLM consistently outperforms traditional baselines on the NCI-ALMANAC dataset, up to a 350% increase in AUPRC on some cell lines. Compared to an oracle model trained on combination data, Bundles matches oracle performance in a zero-shot setting and achieves substantial performance gains when given access even to a fraction of combination data. We further demonstrate that pretraining on single-agent data can bolster combination models, regardless of model architecture.

There are several directions for improvement. One is to further develop our understanding of latent mechanisms in Bundles. Incorporating larger drug-target datasets or information on disease-specific mechanisms could help illuminate further connections between learned latent mechanisms and real biological processes. Extracting this information may require careful and systematic extraction from biology papers.

Another direction is to adapt our work to predicting synergy, which is the focus of many existing methods. While ComboFiLM can be readily applied to any joint property, it is not yet clear how the latent mechanism structure in Bundles relates to synergy. Further investigation is required to leverage therapeutic activity screens, of both single-agents and combinations, for synergy prediction. We anticipate this will be an exciting and key future direction for identifying synergistic combination therapies.
Appendix A

Tables

<table>
<thead>
<tr>
<th>Hyperparameter</th>
<th>Minimum</th>
<th>Maximum</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dropout</td>
<td>0</td>
<td>0.4</td>
</tr>
<tr>
<td>Hidden size</td>
<td>50</td>
<td>400</td>
</tr>
<tr>
<td>Depth of classifier $f$</td>
<td>1</td>
<td>4</td>
</tr>
<tr>
<td>Number of message-passing iterations</td>
<td>2</td>
<td>6</td>
</tr>
</tbody>
</table>

Table A.1: Ranges used for ComboFiLM hyperparameter searches.

<table>
<thead>
<tr>
<th>Split</th>
<th>10%</th>
<th>25%</th>
<th>50%</th>
<th>75%</th>
<th>100%</th>
</tr>
</thead>
<tbody>
<tr>
<td>Training</td>
<td>252</td>
<td>1,728</td>
<td>6,705</td>
<td>16,038</td>
<td>29,796</td>
</tr>
<tr>
<td>Validation</td>
<td>849</td>
<td>1,623</td>
<td>4,092</td>
<td>6,834</td>
<td>7,806</td>
</tr>
<tr>
<td>Test</td>
<td>8,697</td>
<td>7,884</td>
<td>8,562</td>
<td>8,796</td>
<td>9,378</td>
</tr>
</tbody>
</table>

Table A.2: Sizes of training, validation, test sets used in Bundles data-scarce setting.
Table A.3: Model performances in zero-shot setting. Note that Oracle is trained on combination data and is equivalent to MPNN in Table A.4

<table>
<thead>
<tr>
<th>Cell Line</th>
<th>Oracle</th>
<th>Bundles</th>
<th>Random</th>
<th>Single</th>
</tr>
</thead>
<tbody>
<tr>
<td>MALME-3M</td>
<td>0.491 ± 0.144</td>
<td>0.511 ± 0.187</td>
<td>0.352 ± 0.098</td>
<td>0.322 ± 0.069</td>
</tr>
<tr>
<td>PC-3</td>
<td>0.297 ± 0.076</td>
<td>0.357 ± 0.208</td>
<td>0.149 ± 0.011</td>
<td>0.180 ± 0.053</td>
</tr>
<tr>
<td>SNB-19</td>
<td>0.487 ± 0.056</td>
<td>0.258 ± 0.159</td>
<td>0.190 ± 0.080</td>
<td>0.202 ± 0.072</td>
</tr>
<tr>
<td>NCI-H322M</td>
<td>0.211 ± 0.064</td>
<td>0.373 ± 0.165</td>
<td>0.188 ± 0.025</td>
<td>0.173 ± 0.034</td>
</tr>
<tr>
<td>SW-620</td>
<td>0.310 ± 0.126</td>
<td>0.392 ± 0.170</td>
<td>0.187 ± 0.034</td>
<td>0.204 ± 0.038</td>
</tr>
<tr>
<td>ATCC</td>
<td>0.490 ± 0.148</td>
<td>0.350 ± 0.219</td>
<td>0.186 ± 0.060</td>
<td>0.194 ± 0.073</td>
</tr>
<tr>
<td>OVCAR-4</td>
<td>0.207 ± 0.095</td>
<td>0.372 ± 0.132</td>
<td>0.203 ± 0.043</td>
<td>0.195 ± 0.038</td>
</tr>
<tr>
<td>RPMI-8226</td>
<td>0.487 ± 0.105</td>
<td>0.485 ± 0.035</td>
<td>0.468 ± 0.027</td>
<td>0.414 ± 0.020</td>
</tr>
<tr>
<td>ACHN</td>
<td>0.323 ± 0.177</td>
<td>0.306 ± 0.128</td>
<td>0.192 ± 0.052</td>
<td>0.221 ± 0.049</td>
</tr>
</tbody>
</table>

Table A.4: Model performances in data-scarce setting after finetuning on the full combination dataset.

<table>
<thead>
<tr>
<th>Cell Line</th>
<th>MPNN</th>
<th>Bundles</th>
<th>Single</th>
</tr>
</thead>
<tbody>
<tr>
<td>MALME-3M</td>
<td>0.491 ± 0.144</td>
<td>0.746 ± 0.132</td>
<td>0.618 ± 0.182</td>
</tr>
<tr>
<td>PC-3</td>
<td>0.297 ± 0.076</td>
<td>0.579 ± 0.142</td>
<td>0.521 ± 0.169</td>
</tr>
<tr>
<td>SNB-19</td>
<td>0.487 ± 0.056</td>
<td>0.600 ± 0.093</td>
<td>0.508 ± 0.110</td>
</tr>
<tr>
<td>NCI-H322M</td>
<td>0.211 ± 0.064</td>
<td>0.517 ± 0.130</td>
<td>0.355 ± 0.111</td>
</tr>
<tr>
<td>SW-620</td>
<td>0.310 ± 0.126</td>
<td>0.623 ± 0.130</td>
<td>0.518 ± 0.171</td>
</tr>
<tr>
<td>ATCC</td>
<td>0.490 ± 0.148</td>
<td>0.650 ± 0.139</td>
<td>0.541 ± 0.135</td>
</tr>
<tr>
<td>OVCAR-4</td>
<td>0.207 ± 0.095</td>
<td>0.605 ± 0.138</td>
<td>0.488 ± 0.092</td>
</tr>
<tr>
<td>RPMI-8226</td>
<td>0.487 ± 0.105</td>
<td>0.596 ± 0.081</td>
<td>0.658 ± 0.041</td>
</tr>
<tr>
<td>ACHN</td>
<td>0.323 ± 0.177</td>
<td>0.470 ± 0.096</td>
<td>0.466 ± 0.142</td>
</tr>
</tbody>
</table>
Appendix B

Figures

Figure B-1: Distribution of linked SMILES per GO term in the unfiltered version of drug-mechanism data. We want to keep mid-level GO terms in the hierarchy. Based on this distribution, mid-level GO terms are defined as those with 5-11 linked SMILES.
Figure B-2: Tiered performances varying with the percentage of dataset exposed to the model. Points correspond to the median of AUC for the different diseases. Error bars correspond to the upper and lower quartiles.


with combination therapy in hiv-infected adults with cd4 cell counts from 200 to 500 per cubic millimeter. New England Journal of Medicine, 335(15):1081–1090, 1996.


