UNIVERSITY OF AMSTERDAM

MSc ARTIFICIAL INTELLIGENCE
MASTER THESIS

Drug Synergy Prediction with Learned Embeddings

by

EDUARDS SIDOROVIČS
12789429

21st November, 2021

48 ECTS
November 2020 - September 2021

Supervisors:	*Assessor:*
Dr. Efstratios Gavves	Prof. dr. C. I. Sánchez Gutiérrez
Dr. Jonas Teuwen	
Prof. L.F.A. Wessels	

NETHERLANDS CANCER INSTITUTE
ANTONI VAN LEEUWENHOEK

Contents

Abstract

Combination therapy have proved to be very effective to reduce the side effects and to increase efficiency of the cancer treatment. However, potentially negative side effects and high number of drugs and cancer types does not allow to explore all the combinatorial space. Large-scale combination screening data has enabled computational methods to prioritize the potential synergistic candidates for the future tests. Despite recent significant improvement in synergy prediction using deep learning, there is still room for improvement. These models use precomputed molecular fingerprints, however recent research showed that learned molecular representation can achieve better results and generalize better.

In this thesis we investigate how we can improve the anti-cancer drug synergy model by using graph neural networks to obtain molecular representation. We utilize the Direct Message Passing neural network (Yang et al., 2019) to learn molecular embeddings and explore if multi-head attention or Transformer architectures can improve drug representation for the synergy prediction task. We showed that graph neural networks on average performs very similar to the baseline model. Using more advanced node aggregating techniques do help to improve the results on average, especially if we combine models in the ensemble. However these methods do not generalize well on the unseen drugs and cell lines. Finally we present list of potentially synergistic drug combinations which were not yet explored in the literature.

Section 1. Introduction

Cancer is one of the leading causes of death worldwide. Not that long ago treatment was based on the type of cancer with the idea of ”one size fits all”. We have only recently started to see the development of precision oncology (Doherty et al., 2016; Garraway et al., 2013), where treatments are based on the patients' genomic data (Garraway et al., 2013). However cancer is a complex disease and focusing just on a single target might not produce desirable outcomes. Moreover, it is prone to acquired resistance (Housman et al., 2014; Nikolaou et al., 2018). Combination therapy can lead to better efficiency (Csermely et al., 2013; Jia et al., 2009) and can reduce adverse side effects due to decreased dosages (O'Neil et al., 2016). However drug combinations can also have opposite effects (Hecht et al., 2009). It is thus important to identify synergistic drug pairs.

Because of the possible adverse side effects and limited time constraints, trial and error experimentation *in vivo* are feasible only on well-tested drug combinations, which does not guarantee success. For instance, Hecht et al. (2009) performed a clinical trial on patients with metastatic colorectal cancer. The authors tested if the addition of panitumumab to bevacizumab could benefit cancer patients. Results where that progression-free survival (FPS) decreased by 5 months and toxicity was increased. The following example shows the potential pitfalls of the trial and error approach. A safer approach is to test drugs for potential synergistic reactions in high-throughput screening (HTS) (He et al., 2018). This approach usually involves combining 8 different dilution concentrations of two drugs, which results in 8x8 dose-response matrix. 6 drug pairs can be screened at the same time using a 384-well assay plate. After 72 hours, cell viability is measured and is employed to calculate different synergy scores. Despite its simplicity, it is still impossible to test all the drug combinations (Goswami et al., 2015; Morris et al., 2016).

Computational methods can be efficiently employed to help identify potential synergistic pairs and prioritize them for further experiments. Earlier models in drug synergy prediction were using classical machine learning models like Random Forest (Breiman, 2001), Support Vector Machine (Cortes and Vapnik, 1995), and other methods (Bansal et al., 2014; Chen et al., 2013; Li et al., 2015; Zhao et al., 2011). More recent publications have shown that deep learning can significantly improve prediction accuracy (Kuru et al., 2020; Preuer et al., 2018). These models still heavily rely on a prior knowledge such as biological network interactions, omics data (Liu and Xie, 2021) and chemical structure.

Deep learning is proved to be a very flexible method that can learn to extract features from the raw data, which are better than human engineered features (Farabet et al., 2012; Krizhevsky et al., 2012). Similarly, some work has been done to extract features from a molecular graph (Gilmer et al., 2017; Kearnes et al., 2016; Schütt et al., 2017a; Smith et al., 2017) to predict useful properties. However learned feature extractors do not always outperform human engineered ones (Wu et al., 2018; Yang et al., 2019), which is not common in other machine learning domains e.g. natural language processing, computer vision. Reasons for that are that current models are not able to extract good molecular representations and datasets are too small for such data-greedy methodology as deep learning, which is a common problem in the medical domain. However, with the development of HTS methods more data is available. Zagidullin et al. (2019) combined multiple combination screens, standardized and harmonized data to facilitate more research in the area of drug synergies.

Most of the work on drug synergy prediction shows good results of unseen combinations of drugs and cell lines. However, these models do not generalize well for the unseen drugs or cell lines (Liu and Xie, 2021; Preuer et al., 2018), Kuru et al. (2020) do not even report these results. Though such models still can be helpful to find drug combinations which can potentially improve the treatment, it still suffers from a severe limitation: it does not allow to discover new drugs. Analytical drug discovery is still at its very early stages and advancing together with the better models for the drug representation (Wu et al., 2018; Yang et al., 2019). Despite the lack of generalization power of these models, there was a successful attempt for antibiotic prioritization (Stokes et al., 2020).

1.1. Contribution

In this thesis, we investigate how we can improve the anti-cancer drug synergy model. Xia et al. (2018) showed that drug representation is the most contributing factor for the synergy score prediction. The current state of the art model for anti-cancer drug synergy prediction is MatchMaker (Kuru et al., 2020), which uses fixed drug descriptors. However recent work shows that learned representations usually outperforms human designed features for larger datasets (Yang et al., 2019). Hence we want to investigate if we can improve MatchMaker by replacing fixed descriptors with the learned embeddings and see if it also improves generalization for unseen drugs and cell lines.

Current methods for graph representation learning use simple aggregation functions such as mean, sum, norm, etc. We investigate whether technically more advanced approaches such as Multi-Head Attention (MHA) and Transformer (Vaswani et al., 2017) architectures can be used for obtaining better graph representations, hence improve final synergy prediction.

Overall our contributions are summarized as follows:

- we use the current state of the art model for molecular graph embedding to predict synergy score and compare results with models which use human engineered molecular features,
- to obtain graph molecular representation we use multi-head attention and Transformer architectures to test if these methods can improve synergy prediction model,
- we investigate if current methods can generalize well for the unseen drugs or cell lines and if proposed methods can improve it.

1.2. Outline

This thesis consists of 6 sections. In section 2 we explain what is synergy in the drug context and how it can be measured. The related work section discusses recent developments in drug embeddings and synergy prediction models and lays the ground for the section 4, where we introduce our synergy prediction model. Section 5 starts with an explanation of the experimental setup and proceeds with showing the results and analyzing them. We finish section 5 with suggestions for the future research. Finally, we conclude this thesis in the section 6.

Section 2. Preliminaries

The following section provides an introduction to what is drug synergy. Further we introduce two the most popular drug synergy metrics and explain the differences between them.

2.1. Synergy

Drug combination can be classified as synergistic, antagonistic or additive (also called non-interacting in the literature) depending on its observed response in comparison with the expected response, calculated under the assumption that drugs do not interact. If the actual response is better than expected, then this drug pair can be classified as synergistic, and it is classified as antagonistic if the response is lower than expected.

Some of the drug effects are desirable whereas other are better to be avoided. Combination therapy can amplify both desirable and undesirable effects. In the best case scenarios, the goal is to find drug combinations where positive effects are synergistic and negative effects are antagonistic (Tallarida, 2011). Even though synergy is a desirable property, it is not a panacea. It is important to distinguish between *synergy* and *efficacy*. Synergy is a measure of how drugs interact with each other, whilst efficacy measures the phenotypic response of a drug combination. Some of the drug pairs can be highly synergistic, however their phenotypic response is too low to reach therapeutic efficacy (Tang et al., 2015).

Identifying synergistic and antagonistic drug pairs would allows us to nurture our understanding of drug sensitivity and resistance (Yeh et al., 2009). Synergy appears when both drugs target different proteins operating on different pathways, but responsible for the same function (figure 1). Each drug could have little effect individually, since there is an alternative molecular mechanism to exert the same cell function. However if two drugs block the pathways, a joint product cannot be formed anymore. Hence what one drug cannot achieve, is achieved from the combination of drugs. Antagonism appears when two drugs are doing the same job. It can appear if two drugs target essential pathways, as a result, one of the drugs will limit the growth of the essential product and will mask the results of the other drug (figure 1). Another type of antagonistic reaction is when both drugs target the same non-essential linear pathway. For example, the first drug already does the job of completely stopping the flux to the non-essential metabolite, then it completely neutralizes the effect of the second drug. This type of interaction is called co-equality (figure 1).

As was discussed above, a drug interaction is classified depending on the deviation from the expected response (in the literature expected response is usually called additivity). However, there is no consensus on how to measure it. Multiple reference models were proposed, which are based on a set of

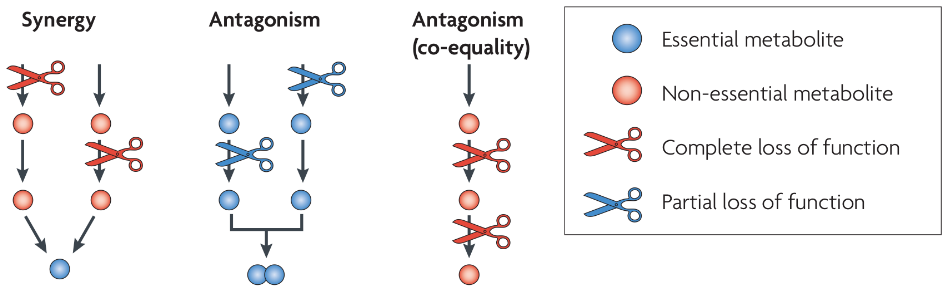
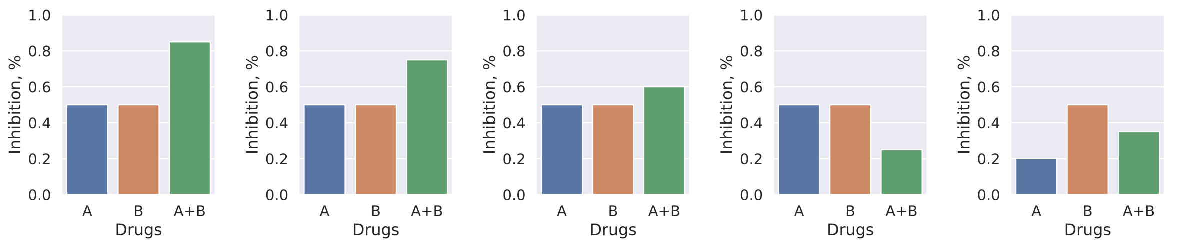


Figure 1: Relation between drug interaction and metabolic networks. Figure from Yeh et al. (2009)



(a) Synergy **(b)** Additivity **(c)** Antagonism **(d)** Antagonism (reciprocal suppres.) **(e)** Antagonism (directional suppres.)

Figure 2: Overview of the drug interaction types assuming Bliss model for calculating additivity

different empirical and biological assumptions. However, due to a lack of precise knowledge of the mechanisms of actions, it is hard to validate these models a priori (Tang et al., 2015).

The most common metrics to measure the expected drug combination effect is the Bliss independence and Loewe additivity model. Both of these models are dose dependent, which means that drug synergy is not the only property of a combination of two drugs, but also depends on the dose of these drugs.

2.1.1 Bliss Independence Model

Bliss Independence model (Bliss, 1939) assumes that two drugs are independent of each other which gives the following formulation of the expected response:

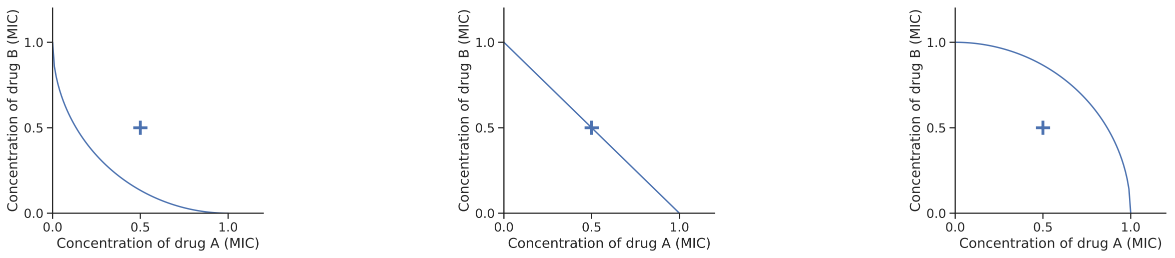
$$y_{\text{Bliss}} = y_1 + (1 - y_1)y_2 = y_1 + y_2 - y_1 y_2 \tag{1}$$

For example, drugs A and B cause 50% cancer cell growth inhibition. If the drugs do not interact (additivity) their combination will cause 75% growth inhibition (figure 2b). If observed growth inhibition is higher then this drug combination is synergistic (figure 2a), if lower than 75%, then it is antagonistic (figure 2c). A more drastic example of antagonism is hyper-antagonism or suppression (Yeh et al., 2009), where the combined effect of drug combination is lower than one of the drugs alone (figure 2e) or both (figure 2d). Such cases are not common, but not very rare either. Directional suppression appears in the case when one perturbation masks perturbation in the other drug. Reciprocal suppression is not well understood, hence literature lacks specific reasons for such reaction (Yeh et al., 2009). Though it might be potentially related to the Eagle effect (Eagle and Musselman, 1948), which states that increased concentration of some drugs can decrease their effect.

2.1.2 Loewe Score

To measure additivity Loewe (1928) suggested assuming that the drug does not interact with itself. So if drug A does not interact with drug B, then using their combination should give the same effect as using the double concentration of A or B. More formally, we have to know dose-response relations of both drugs. Lets assume that dose-response function for drug A is $y = f_A(x)$ and for drug B $y = f_B(x)$, where y is cell line growth inhibition and x is a dose. Both drugs can achieve y_{Loewe} growth inhibition. For drugs A and B to be non-interacting it has to satisfy the following equation (Tang et al., 2015):

$$\frac{x_A}{f_A^{-1}(y_{\text{Loewe}})} + \frac{x_B}{f_B^{-1}(y_{\text{Loewe}})} = 1 \tag{2}$$



(a) Synergy **(b)** Additivity **(c)** Antagonism

Figure 3: Overview of the Loewe model isoboles for different drug interaction types

where x_A, x_B are drug dosages and $f_A^{-1}(y)$, $f_B^{-1}(y)$ are inverse mapping of response y to dose x. The reasoning behind this formula comes from the fact that if we combine two identical drugs, then $f_A(x) = f_B(x)$. We will be able to derive that $y_{\text{Loewe}} = f(x_A + x_B)$, which supports the idea that if drugs are non-interacting they should then produce the same response as a combination of the same drug.

A more visual explanation of the Loewe model can be seen in figure 3. Lines on these graphs represent constant inhibition (isobole). For example, if we take 0.5 minimum inhibitory concentration (minimum required concentration which would prevent cell growth) (MIC) of drug A and 0.5 MIC of drug B and combine them we should expect the inhibition as if we use 1 MIC of drug A or B 3b. Linear line represents that every dosage of drug A can always be replaced with the constant increase of dosage B, which represents additivity or non-interaction. In figure 3a we see that isobole is convex. Meaning that to achieve the inhibition equivalent to the full dosage of A we can use a fraction of drug A and a fraction of drug B, hence, as a result, a much less dosage. On contrary, in figure 3c we see an antagonistic reaction, where to achieve the same inhibition as with only one drug, for the combination of A and B we have to increase the dosage.

2.2. Loewe vs Bliss

It was noted that these two metrics sometimes lead to opposite conclusions (Kuru et al., 2020; Zagidullin et al., 2019). Such discrepancies are the result of different model assumptions. However, the Bliss score is expected to work better if drugs are targeting separated pathways, but the Loewe score is more accurate if drugs target similar targets or pathways (Tang et al., 2015). Nevertheless, the Loewe score is considered by many as a gold standard for non-interaction (Twarog et al., 2020) and most of the publications on synergy prediction are using the Loewe score.

Section 3. Related Work

The following section reviews the most recent work on predicting synergies and molecular encoding. Specifically, in section 3.1 we review the most important models for the synergy prediction task such as DeepSynergy (Preuer et al., 2018), TranSynergy (Liu and Xie, 2021), MatchMaker (Kuru et al., 2020) and analyze their differences. In section 3.2 we review different ways to encode molecules, and in general terms describe algorithm behind the current state-of-the-art molecular encoding model Direct Message Passing Neural Network (DMPNN) (Yang et al., 2019).

3.1. Predicting Drug Synergies

In 2016 O'Neil et al. published a large synergy dataset with over 2 000 synergy measurements, which were computed on 38 different drug combinations and 39 cancer cell lines. This dataset opened doors for more data-greedy methods such as deep learning. One of the first models which investigated the power of deep learning in the drug synergy prediction was DeepSynergy (Preuer et al., 2018). The authors completely ignored data on the drug response and focused solely on the synergy score prediction. DeepSynergy was trained on the dataset by O'Neil et al. (2016). Authors compared DeepSynergy with state-of-the-art machine learning methods such as Random Forests (Breiman, 2001), SVM(Cortes and Vapnik, 1995), Elastic Nets (Friedman et al., 2010) and Gradient Boosting (Friedman, 2001).

For drug representation Preuer et al. (2018) used 3 different types of chemical features:

1. counts of extended connectivity fingerprints with a radius 6 (ECFP6) (Rogers and Hahn, 2010),
2. physico-chemical properties, calculated by ChemoPy (Cao et al., 2013),
3. a binary indicator of toxic substructures.

For the cell line features, the authors used gene expression profiles from the ArrayExpress database (Iorio et al., 2016) and performed Factor Analysis for Robust Microarray Summarization (FARMS) (Hochreiter et al., 2006). DeepSynergy model consists of two fully connected layers followed by a single output neuron. It uses as input 2 concatenated drug feature's with cell line features. DeepSynergy significantly outperforms all other methods on unseen drug pairs. However, authors note that the method performs poorly on unseen drugs and cell lines, attributing this issue to the limitation in the size and diversity of the dataset.

Kuru et al. (2020) proposed another model, MatchMaker, which significantly outperforms DeepSynergy. For their experiments, the authors used the DrugComb database (Zagidullin et al., 2019), which contains 466 033 combinations across 112 cell lines and 4 150 drugs. Similarly to Preuer et al. (2018) in MatchMaker cell lines were represented by gene expression data from Iorio et al. (2016) and drugs were represented by chemical descriptors obtained from ChemoPy library.

The main differences lie in the choice of the architecture and loss function. MatchMaker first encodes concatenated representation of each drug with a cell line (Drug Specific Subnetwork, DSN), then the output of these two combinations is concatenated and is used as an input for the second sub-network to obtain the Loewe score (Synergy Prediction Network, SPN). The authors pointed out that synergistic drug pairs are much more important to get right since they have therapeutic potential. Therefore the authors use weighted mean squared error (MSE) loss, where weights were calculated as a difference between true Loewe score and the minimum of all Loewe scores in the training data, thus increasing the loss for bad predictions in the synergistic pairs.

MatchMaker outperforms DeepSynergy in all 3 measured performance metrics. MatchMaker achieves MSE value 79.49, Pearson correlation 0.79, and Spearman correlation 0.74, while DeepSynergy 112.6 (MSE), 0.69 (Pearson) and 0.61 (Spearman). However, it is unclear if improvement comes from parallel architecture, deeper architecture or the weighted loss function.

Since the main goal of *in silico* synergy prediction is to prioritize the candidates for *in vitro* experimentation, it is important to be more correct at the top of the predicted synergy scores. To evaluate it Kuru et al. transformed regression task to a binary classification task (synergistic or not synergistic drug combination) and evaluated partial AUC (pAUC), which considered samples up to 10% false-positive rate range. According to this metric MatchMaker also outperformed second-best competitor DeepSynergy, 0.97 and 0.92 pAUC score respectively.

Kuru et al. (2020) investigated the drug pair combinations, which were consistently predicted as synergistic, however, the measured Loewe score indicates that the pair is antagonistic. For such pairs the authors also looked at Bliss independence, Highest single agent, Zero interaction potency and S synergy scores. For most of the pairs, the majority of the metrics agreed that drugs are synergistic, even though the Loewe score is negative. These results point out differences in the assumptions behind every model and show the limitation of usage of a single score to make a conclusion about the synergistic effect.

However, the main drawback of the above-mentioned methods is the lack of interpretability. It is hard to establish any relationship between drug chemical properties and its mechanism of action on a specific cell. Liu and Xie (2021) proposed a model TranSynergy, where they use drug-target interaction data for the drug representation. This step already makes the model more interpretable. Furthermore, they developed a method called *Shapley Additive Gene Set Enrichment Analysis* (SA-GSEA) to analyse which genes are contributing to the synergistic drug combination.

Liu and Xie (2021) also used another cell line representation. Apart from gene expression they used gene dependency data (McFarland et al., 2018), which represents genes that code for the most essential proteins for a cell line (drugs which target essential proteins are more likely to cause serious damage to the cell). TranSynergy was consistently performing better with gene dependency data, but the best results were obtained when Liu and Xie (2021) used both gene expression and gene dependency.

Apart from model input, TranSynergy differs in architecture. It uses modified Transformers (Vaswani et al., 2017). Liu and Xie (2021) removed the positional encoding, since it is not relevant for this problem. The model consists of 3 main components: (1) dimension reduction, (2) attention mechanism and (3) synergy score predictor.

TranSynergy model was trained on the screen produced by O'Neil et al. (2016). As a baseline model they used DeepSynergy. Liu and Xie (2021) showed that TranSynergy was outperforming DeepSynergy in all 3 measured metrics: MSE dropped from 243 to 231, Spearman correlation and Pearson correlations increased from 0.698 and 0.726 to 0.730 and 0.746 respectively. Even if TranSynergy uses the same representation of cell lines (only gene expression data), it still outperforms DeepSynergy, which shows the added value of the novel drug representation and attention-based architecture.

Liu and Xie (2021) also tested if adding extra chemical information of the drugs can improve the model performance. First, they tried to use features that were used in DeepSynergy, namely physiochemical, toxophores and fingerprint information. For the second experiment they used graph convolutional neural network to extract the fingerprint from the drug chemical structure. In both cases TranSynergy did not benefit from extra information, on contrary, its performance slightly deteriorated.

To predict synergistic drug pairs Xia et al. (2018) built a model, which would predict drug combination response. Such a model can explain 0.94 of the response variance ($R^2 = 0.94$). For drug representation authors used Dragon7 software (Mauri et al., 2006). This is commercial software, which generates features based on multiple molecular descriptors, e.g. ring descriptors, topological indices, path counts, etc. However, they could use only 54 of the 104 drugs available in NCI-ALMANAC data (Holbeck et al., 2017), because they were not available in the Dragon software. For the cell line representation, they used gene expression (gene transcripts expression were taken from CellMiner (Reinhold et al., 2012)), microRNA expression (also downloaded from CellMiner) and protein abundance (from Proteome Database (Gholami et al., 2013)).

Before concatenation of all the features, authors use separate fully connected layers to reduce dimensions for the cell line features, and one network for both drugs encoding. It is followed by another sub-model, which consists of 4 fully connected layers with residual connections, which outputs predicted percentage growth. Xia et al. (2018) obtain encouraging results, and most of the prediction power comes from drug features - the difference between using one-hot encoding and Dragon descriptors results in 0.81 R^2 score improvement. The authors also showed that gene expressions are the most effective features to represent cell line, adding microRNA and proteomics data improve final R^2 score only slightly from 0.936 to 0.944. However, the authors did not have a hold out test set and did not test their model on unseen drugs or cell lines.

3.2. Molecular Encoding

There are two main lines of research on drug representation. The first one is expert-engineered descriptors and fingerprints. This method requires domain expertise to improve information representation in the embeddings. An alternative approach is to optimize the model architecture to extract better features for the specific downstream task. Such models try to extract relevant information from less processed data such as SMILES or molecule graph.

Molecular fingerprints and descriptors One of the most popular molecular fingerprints is the Morgan fingerprint (Morgan, 1965). It uses hashing algorithm to capture molecular structure and wraps it up in 2048 binary vector, where each bit represents the presence or absence of particular substructure of the molecule. Capecchi et al. (2020) notes that Morgan fingerprint has a poor perception of the global features such as size and shape, hence its performance can be bad for larger molecules. Later Rogers and Hahn (2010) used this algorithm to create extended-connectivity fingerprints (ECFP). The main difference between these two algorithms is that ECFP runs for a predefined number of iterations while the Morgan algorithm runs until it achieves a unique identifier. Also, ECFP is more computationally scalable with a much more effective hashing scheme. Other popular alternatives also include global molecule features in the representation, e.g. Dragon descriptors (Mauri et al., 2006), RDKit descriptors (Landrum, 2013). Most of the research in this field focuses on domain expertise to create better feature representation (Yang et al., 2019), some of the publications also use 3D atomic coordinates to produce a better representation (Feinberg et al., 2018; Schütt et al., 2017b).

Single agent response Gayvert et al. (2017) used a small combinatorial screen on 40 drugs to predict if drugs are synergistic using only single agent dose response. Single drug was represented as a percentage of concentration required to inhibit 50% of the growth of 27 tested cell lines. The authors used Random Forest (Breiman, 2001) algorithm on the concatenated features of two drugs and achieved AUC of 87%. This approach does not leverage any background information about drug molecular structure and cell gene expression. These results indicate that even the single agent dose response can provide performance benefits when predicting synergistic responses.

SMILES SMILES is a unique representation of the molecule, hence there were some attempts to learn representation from it. Mayr et al. (2018) conducted a large-scale comparison of different machine learning methods to predict drug target interactions. They have used fixed drug representation and tested different machine learning approaches, i.e. deep learning, Random Forest (Breiman, 2001), Support Vector Machine (Cortes and Vapnik, 1995), k-Nearest Neighbours (Fix and Hodges, 1989). Apart from using drug fingerprints they also tried to predict drug-target interaction with drug's SMILES representation. Treating SMILES as a string input Mayr et al. (2018) used Recurrent Neural Networks (RNN), specifically Long short-term memory (LSTM) architecture (Hochreiter and Schmidhuber, 1997), to make predictions. However, this method did not perform that well, simple deep learning architecture consisting only of fully connected layers and SVM applied on fixed descriptors were significantly better than proposed model. Wu et al. (2018) suggests that SMILES lack information about electronic or topological features of molecules, and under limited amount of data these methods will not be able to outperform other methods which provides more information.

Drug-Target There are multiple ways how a drug can be represented in order to extract useful features for a downstream task. Liu and Xie (2021) avoided using drug chemical properties to ensure the model's interpretability. Instead, Liu and Xie (2021) used drug-target and protein-protein data to get drug representation. Drug-target data was used to create a binary vector, to represent downstream non target protein, they applied random walk with restart algorithm on a protein-protein interaction (PPI) graph. The authors showed that PPI information is crucial for good drug representation since the presence of these features significantly improves the predictions power. Liu and Xie (2021) admit that using drug target information limits the application of the TranSynergy model, since chemotherapy drugs do not have a specific target and drug-target data is much more limited. The authors mention that different methods for the prediction of drug targets can be used, however, they did not provide any evidence that such data can yield comparable results.

Molecular Graphs A more natural way to represent a molecule is a graph. By defining atoms as nodes and bonds as edges, we can utilize graph structure to get better molecular representations. There are multiple ways how one can extract features from a graph, which is reflected in different graph based models. Duvenaud et al. (2015) proposed a method, called Graph Convolutions (GC), which is similar to ECFP approach to gradually aggregate information from neighbouring nodes. However, the authors replaced the fixed hash function by neural networks, which enables learning a function to extract useful features for the downstream task. Generally, GC works on the undirected graph and uses the same function for a node and its neighbours. Later Gilmer et al. (2017) proposed message passing neural network (MPNN), which is a generalized framework for most of the graph based models. MPNN model operates in 2 phases: message passing and readout. The message passing phase consists of T steps, where each node aggregates messages from the neighbouring nodes. Messages are computed by abstract message function, which takes atoms' features and bond features as an input. The readout phase aggregates information of the whole graph and maps these features to the graph properties.

There are other popular variations of graph networks, i.e. directed acyclic graph, deep tensor neural networks (DTNN)(Schütt et al., 2017a), ANI-1 (Smith et al., 2017), Weave (Kearnes et al., 2016). These approaches are illustrated and briefly summarized in the figure 4.

All of these methods were introduced in different contexts and there was a lack of clarity on which model is better. Wu et al. (2018) tried to organize this research field and created a benchmark for molecular machine learning. In their work, the authors compared different fingerprints and descriptors to different graph based models (figure 4 provides an overview of the graph based models tested by Wu et al. (2018)) on 17 different datasets from different categories: quantum mechanics, physical chemistry, biophysics and physiology.

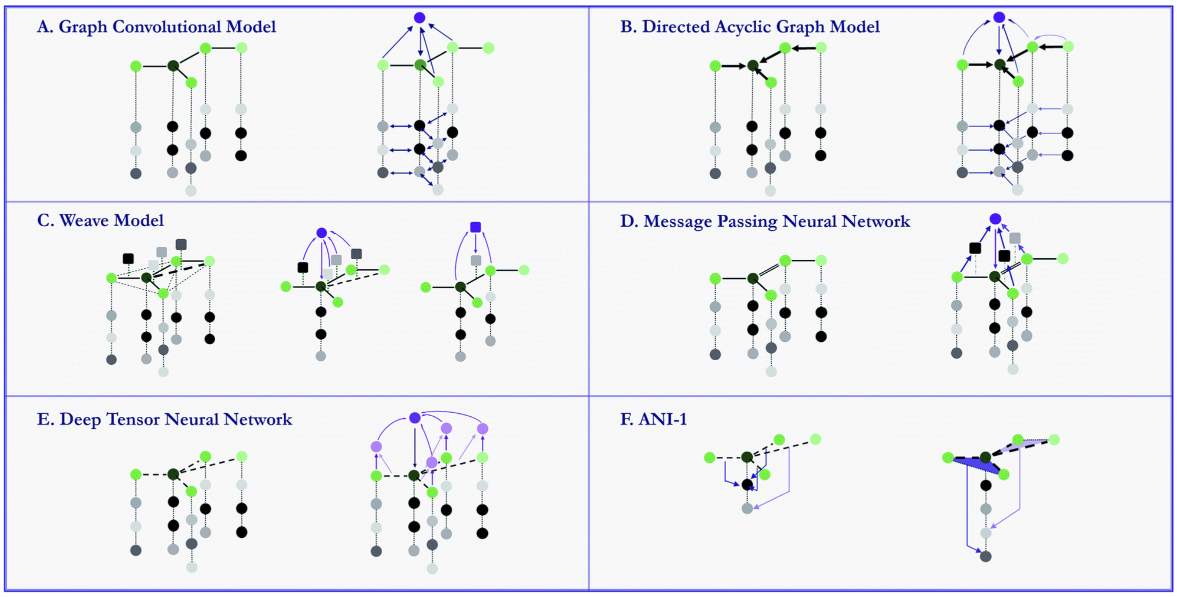

A. Graph Convolutional Model
B. Directed Acyclic Graph Model
C. Weave Model
D. Message Passing Neural Network
E. Deep Tensor Neural Network
F. ANI-1

Figure 4: Variations of graph based models for molecular representation. In this illustration, the aim is to build a representation of the central dark green atom. (A) Graph Convolutional Model. Dark green atom aggregates information from connected neighbouring nodes. (B) Directed Acyclic Graph Model. All bonds are directed to the central node, and features are propagated to update from graph leaf nodes to the centre. (C) Weave Model. Atom features are updated by aggregating information from all the nodes (including non connected), also bond features are updated by using the information of both atoms connected by respective bond. (D) Message Passing Neural Network. Every neighbouring atom creates a message using its and bond features, afterwards the central atom aggregates all the messages to update its representation. (E) Deep Tensor Neural Network. Features of the atom are updated based on all the other nodes and their physical distance. (F) ANI-1. Accurate Neural network engine for molecular energies. For feature construction it uses information about the distance of all the nodes and angular information between two neighbouring nodes. Figure from Wu et al. (2018)

Wu et al. (2018) showed that there is no clear evidence that learned molecule embeddings are outperforming pre-computed fingerprints or descriptors, however notes that graph neural networks struggle to deal with downstream tasks under highly imbalanced datasets and data scarcity. Later Yang et al. (2019) tried to estimate this threshold, where one should choose learned representations over the fingerprints. The authors show that usually on datasets with less than 1 000 molecules fingerprints models perform better, because on the smaller datasets MPNN models fail to identify and extract relevant features.

Even though learned molecule representations usually perform better on bigger datasets, most of the MPNN algorithms use less message passing steps than the diameter of the biggest molecule in the dataset, hence some atoms will never receive a message from the other side of the graph. It results in representation that fails to capture the global features of the molecule. To tackle this problem and also the problem of data scarcity, Yang et al. (2019) proposes to use hybrid representation which consists of learned representation and fixed descriptors. On most of the datasets hybrid representation yielded better results than other representations, however on some datasets addition of RDKit features only harms the performance. The authors suggest that it is worth using hybrid representation when the dataset is small or RDKit features are particularly relevant for the specific task.

Apart from experimenting with the different molecule representation types, Yang et al. (2019) also proposed improvements for the MPNN model, which they build on top of Gilmer et al. (2017) work. The main difference comes from the type of messages that are passed in the first phase. It is proposed

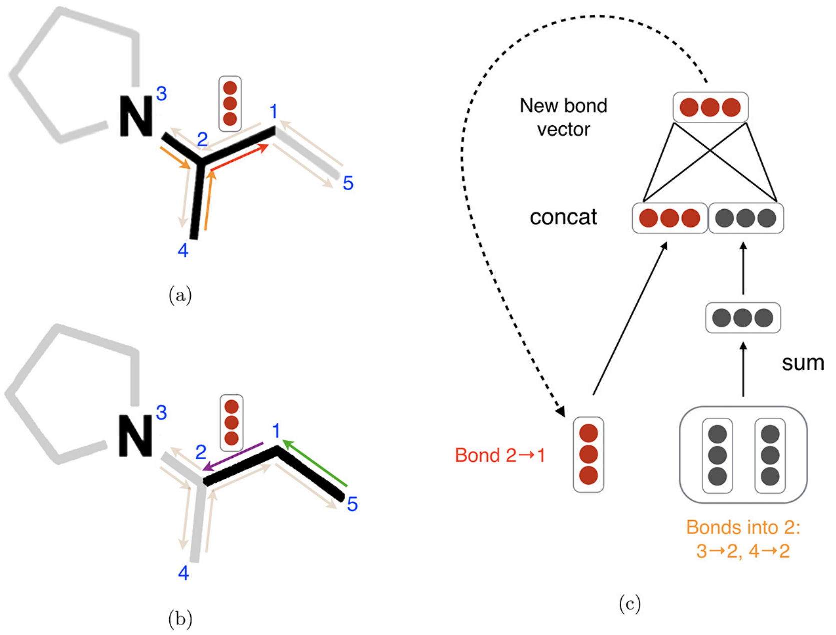


Figure 5: Visualization of DMPNN model. (1) To construct message $m_{2\to1}$ only red hidden states are used. Other hidden states, e.g. $h_{5\to1}$, are ignored. (b) Similarly if we want to construct $m_{1\to2}$ we use information only from the green bond. (c) Illustrates how the update function works for (a) example. Figure from Yang et al. (2019)

to use directed edges for message building (hence the name of the model Directed Message Passing - DMPNN). Such design choice reduces loops, hence minimizes noise in the graph representation. For example in figure 5, message 1→2 would be passed to nodes 3 and 4, but not to node 1. In the MPNN this message would be passed also to 1.

Generally, DMPNN model operates on the directed graph $G(V, E)$ with a set of nodes V and edges E. Similarly to MPNN the process of obtaining the graph G representation is split in two phases: (1) message passing and (2) readout phase. Message passing process consists of T steps, during which at each step t hidden state of every node h_v^t is updated with a message m_v^t. DMPNN model produces the message based on directed bond features. In the context of molecular structure, where the graph is not directed, we can convert every edge to two directed edges. Formally message passing phase can be expressed with the following equations:

$$h_{vw}^0 = \tau(W_i[x_v, e_{vw}]) \tag{3}$$

$$m_{vw}^{t+1} = \sum_{k \in N(v) \setminus \{w\}} M_t(x_v, x_k, h_{kv}^t) \tag{4}$$

$$h_{vw}^{t+1} = U_t(h_{vw}^t, m_{vw}^{t+1}) \tag{5}$$

Where W_i are learned weights. $[\cdot]$ denotes vector concatenation.

The hidden state of the bond h_{vw}^0 is initialized by the linear layer applied on the concatenated input features of the node x_v and edge features e_{vw}, which is passed to activation function τ. The set $N(v)$ contains all neighbours of the node v. To get a new message m_{vw}^{t+1} we use function M_t and apply it on all the neighbouring nodes of v, which are represented by $N(v)$, excluding node w. Function M_t depends on the features of the connected nodes and the current hidden representation of all the incoming edges apart from the reversed message m_{wv}^t. Afterwards we update the current hidden state with the function U_t which takes the current hidden state and received message as an input. Functions M_t and U_t are design choice.

Message passing is repeated for T steps. Afterwards the atom hidden states are obtained in the following way:

$$m_v = \sum_{w \in N(v)} h_{vw}^T \tag{6}$$

$$h_v = \tau(W_a[x_v, m_v])) \tag{7}$$

the final message is the summation of all the incoming edges. We concatenate this message with the node features and multiply with learned weights W_a, afterwards apply nonlinear function τ. To obtain a graph representation, Yang et al. (2019) calculates mean of all the atom hidden states.

$$h = \frac{\sum_{v \in G} h_v}{|V|} \tag{8}$$

Where $|V|$ is number of nodes in the graph G.

Yang et al. (2019) conducted experiments on 19 public and 16 proprietary data sets across different domains and dataset sizes. DMPNN achieved the state of the art results on 12 public datasets and all proprietary datasets. On the other 7 public datasets there is no clear baseline model, which would consistently show better performance on all the these datasets. Conducted experiments were tested on a scaffold split, which the authors showed is a good proxy for a chronological split. The results that follow show that DMPNN model has a much better generalization. Later Stokes et al. (2020) successfully used DMPNN for prioritizing new antibiotics, for further testing *in vitro*.

Section 4. Methodology

The review of related works showed that learned molecule representations consistently matches and outperform state of the art models with fixed molecular descriptors or fingerprints, also they have better generalization power. However in the synergy prediction task previous research mostly used fixed descriptors. Therefore we present a model which predicts synergy scores based on the learned molecule embeddings from the molecular graph. This model can be divided in two sub-models: (1) sub-model to obtain drug representation and (2) sub-model to predict synergy score. In the subsection 4.1 we explain models which serve as a background and elaborate on the additions of the current model to obtain drug representation. In the subsection 4.2 we explain the architecture of the synergy prediction model.

4.1. Drug representation

As was shown in the related work section, D-MPNN (Yang et al., 2019) is the state of the art model for molecular representation. Therefore in this thesis, we use this model as a backbone and apply some changes to it. In this section I elaborate on the design choices for the message passing phase and explain what changes were made in obtaining the graph representation.

4.1.1 Message passing

D-MPNN requires to define a message function M_t and an edge update function U_t. Due to success of Yang et al. (2019) to match or outperform most of the state of the art models, we choose the same simple message passing and update functions:

$$M_t(x_v, x_w, h_{vw}^t) = h_{vw}^t \tag{9}$$

$$\begin{aligned} U_t(h_{vw}^t, m_{vw}^{t+1}) &= U(h_{vw}^t, m_{vw}^{t+1}) \\ &= \tau(h_{vw}^0 + W_h m_{vw}^{t+1} + b_h)) \end{aligned} \tag{10}$$

where $W_h \in \mathbb{R}^{h \times h}$ and $b_h \in \mathbb{R}^h$ are the learned weights, h_{vw}^t is the edge hidden state at step t, x_v and x_w are features of the both nodes of the edge, and τ is activation function.

To obtain hidden atom representation we follow the following steps:

$$h_{vw}^0 = \tau(W_i e_{vw} + b_i) \tag{11}$$

We initialize the edge hidden state by applying a linear layer and an activation function τ on the molecule bond features, which are specified in table 2.

$$m_{vw}^{t+1} = \sum_{k \in \{N(v) \setminus w\}} h_{kv}^t \tag{12}$$

$$h_{vw}^{t+1} = \mathrm{Dropout}(\tau(h_{vw}^0 + W_h m_{vw}^{t+1} + b_h)) \tag{13}$$

A message from edge $v \to w$ is the sum of all incoming edges except the edge $w \to v$ (eq. 12). Lastly, to update the hidden state we apply a linear layer on the new message and sum it with the initial hidden state (eq. 13). And to reduce overfitting we use a Dropout (Srivastava et al., 2014) layer. These two steps are repeated T times. Followed by

$$m_v = \sum_{w \in N(v)} h_{vw}^T \tag{14}$$

$$h_v = \mathrm{Dropout}(\tau(W_o[x_v, m_v] + b_o)) \tag{15}$$

Table 1: Description of the atom features used in the model.

Feature	**Description**	**Size**
Atom	One-hot encoding of the atom type, represented by atomic number	100
Degree	One-hot encoding of the degree of an atom - number of directly-bonded neighbours.	6
Formal Charge	One-hot encoding of the formal charge assigned to an atom	5
Chiral Tag	One-hot encoding of the tag: Tetrahedral CCW/CW, unspecified or other	4
Number of Hs	One-hot encoding of the number of bonded hydrogens	5
Is aromatic	Boolean: if atom is a part of aromatic system	1
Mass	Atomic mass scaled by 0.01	1

Table 2: Description of the bond features used in the model.

Feature	**Description**	**Size**
Type	One hot encoding of the bond type: single, double, triple aromatic	4
Conjugated	Boolean: if bond is conjugated	1
In ring	Boolean: if bond is part of a ring	1
Bond Stereo	One-hot encoding of the bond stereo type: any, E, none, Z, Trans, cis	6

W_i, b_i, W_o and b_o are the learned weights. x_v are atom features listed in tables 1. Atom and bond features are computed with the RDKit package (Landrum, 2013).

4.1.2 Graph representation

In this thesis we explored three options of obtaining the graph representation from computed hidden atom representations: (1) mean of all the hidden states, (2) a weighted sum using multi-head attention (Vaswani et al., 2017) and (3) encoding hidden states with transformers (Vaswani et al., 2017).

Mean. Despite its simplicity, averaging all atoms hidden states has shown good performance and good generalizing power (Stokes et al., 2020; Yang et al., 2019). In this thesis, we used mean as a benchmark to compare proposed changes. Hence we obtain graph representation in the following way:

$$h = \frac{1}{|V|} \sum_{v \in G} h_v \tag{16}$$

where $|V|$ is the number of atoms in the molecule G.

Attention There are different definitions of the attention term. In this thesis we define it as it was used by Vaswani et al. (2017). An attention function is used to compute a weighted sum of the given values, where weights are computed by the compatibility between provided query and keys. Formally this attention mechanism can be written the following way:

$$\begin{aligned} \alpha_i &= \frac{\exp(f_a(\text{key}_i, \text{query}_i))}{\sum_j \exp(f_a(\text{key}_j, \text{query}_i))} \\ &= \textit{Softmax}(f_a(\text{key}_i, \text{query}_i)) \end{aligned} \tag{17}$$

$$out = \sum_i \alpha_i \cdot \text{value}_i \tag{18}$$

where key_i, query_i and value_i are represented as vectors and f_a is compatibility function.

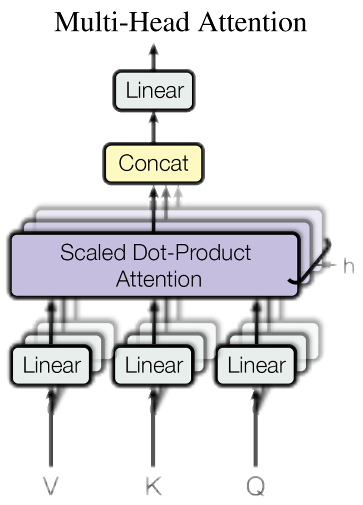


Figure 6: Visualization of Multi-Head Attention. Figure from Vaswani et al. (2017)

Scaled Dot-Product Attention To compute the weights we need to evaluate the compatibility of the query and key pair. Bahdanau et al. (2014) proposed to use feed forward neural network, a popular alternative to it is to use dot-product of two matrices, however in case of the high dimensionality of a query and a key, variance of dot product grows, which may cause to push the output of softmax to be close to zero for significant number of variables, which in turn cause very small gradients, hence it will slow down the training. To overcome this issue Vaswani et al. (2017) proposed to use scale dot product:

$$attention(Q, K, V) = Softmax(\frac{QK^T}{\sqrt{d_k}})V \tag{19}$$

where $Q \in \mathbb{R}^{N_Q \times d_k}$, $K \in \mathbb{R}^{N_i \times d_k}$, $V \in \mathbb{R}^{N_i \times d_v}$, N_Q is number of queries, N_i is number of keys and values, d_k is the number of dimensions of queries and keys, d_v is the number of dimensions of the values.

Multi-Head Attention (MHA) Vaswani et al. (2017) showed that it is more beneficial to map input triplets (query, key, value) to s sub-queries, sub-keys with dimensionality of $d'_k = \frac{d_k}{s}$, and sub-values with dimensionality of $d'_v = \frac{d_v}{s}$. Such architecture allow to compute attention in parallel for s triplets. This will result with s output vectors, which are concatenated together and projected to form the output of the MHA. The MHA computation graph is shown in figure 6 and formulated in equation 20

$$MHA(Q, K, V) = [head_1, \ldots, head_s]W^O \tag{20}$$
$$\text{where } head_i = attention(QW_i^Q, KW_i^K, VW_i^V) \tag{21}$$

Where $W_i^Q \in \mathbb{R}^{d_k \times d'_k}$, $W_i^K \in \mathbb{R}^{d_k \times d'_k}$, $W_i^V \in \mathbb{R}^{d_v \times v'_k}$ and $W^O \in \mathbb{R}^{d_v \times d_v}$.

In the original setting MHA (Vaswani et al., 2017) was used with a self-attention, meaning that queries, values and keys were projected from the same input values. In this thesis, we aim to get a better representation of the molecule graph, where the properties of the drug depend on the other drug and a cell line. Therefore, to compute attention we map queries from the concatenated features of both drugs global features and a cell line representation. Hence queries, keys and values are obtained in the following way:

$$Q = W_q \tau(\text{Dropout}(W_0[D_1, D_2, Cell])) \tag{22}$$

$$K = W_k(H) \tag{23}$$

$$V = W_v(H) \tag{24}$$

where $D_1, D_2 \in \mathbb{R}^{200}$ are chemical descriptors extracted from the RDKit package, $Cell$ is cell line representation, which is represented by the gene expression data from Iorio et al. (2016) and $H \in \mathbb{R}^{N_{\text{atoms}} \times d_h}$ is a matrix of all the computed atom hidden states of one of the input drugs.

Transformers Originally Transformer architecture was designed for machine translation tasks. Hence it has encoder-decoder architecture. However, in the current setting of getting the drug embedding, we are interested only in the encoder part. Figure 7a shows the architecture of the encoder.

The Transformer encoder typically starts with the positional encoding block. In the context of aggregating nodes of the graph, there is no sequence, therefore in our encoder implementation we will ignore positional encoding.

Positional encoding is followed by N identical stacked blocks, where N is a hyperparameter. Each block consists of two sub-layers. The first layer consists of the MHA block, followed by a residual connection (He et al., 2016) and a layer normalization (Ba et al., 2016). The second sub-layer consists of the feed forward network (FFN) which is also followed by a residual connection and a layer normalization. Such block can be represented in the following way:

$$out_1 = \text{LayerNorm}(H + \text{MHA}(Q, K, V)) \tag{25}$$

$$out_2 = \text{LayerNorm}(out_1 + \text{FFN}(out_1)) \tag{26}$$

where Q, K and V are the linearly transformed input features H, which was explained above.

Residual connections are important for the Transformers for multiple reasons. Firstly, Transformers are made to be stacked, and to enable gradient flow for a deep architecture residual connections are one of the solutions (He et al., 2016). Secondly, residual connection helps to preserve the information about the input sequence, which is important for the NLP applications.

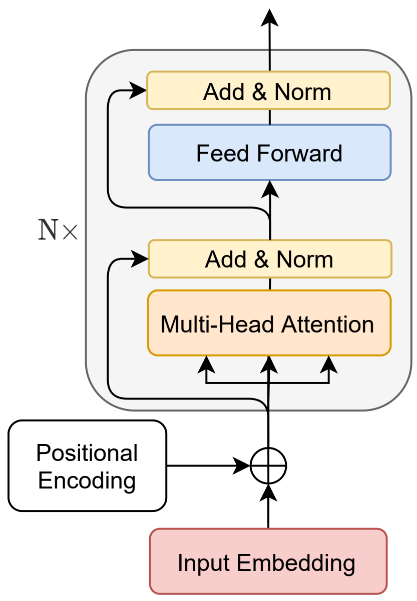
Layer normalization introduced by Ba et al. (2016) reduces the training time and provides small regularization. The main purpose of the normalization is to scale features to be in the same magnitude as the elements in the sequence. Even though Batch normalization is a popular alternative, in the NLP setting it was shown to perform worse and it depends on the mini-batch size, which is typically small with Transformers.

In this thesis, we modified the original Transformer encoder architecture. Firstly, similarly as with the MHA we do not use self-attention, instead, we provide both drugs global features and cell line features. Secondly, in the original transformer architecture Q (Multi-head Attention sub-module input) is calculated from the output of the previous Transformer block, in our architecture Q is consistent among all the blocks. To calculate inputs for the MHA we use the same equations as in 22, 23, 24. Proposed encoder architecture can be seen in the figure 7b.

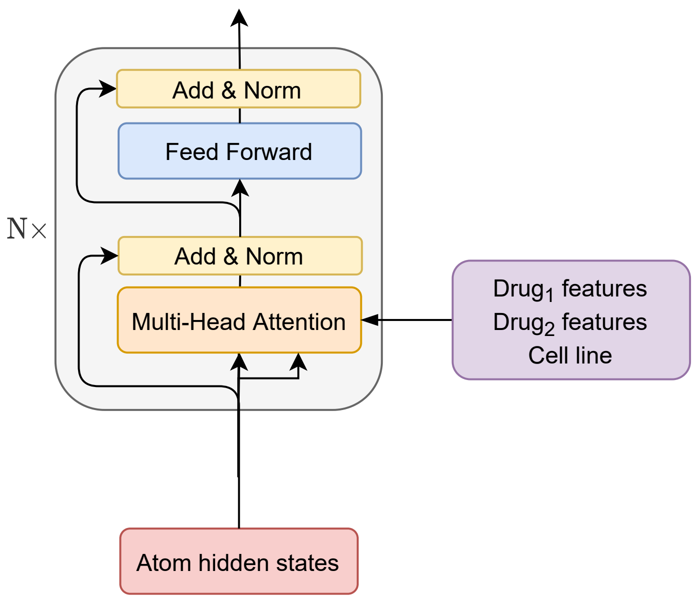
A feed forward sub-module consists of two linear layers with Rectified Linear Unit activation function (Nair and Hinton, 2010)(*ReLU*) and a Dropout layer between them.

4.2. Synergy prediction

Synergy prediction task is modelled as a regression task to predict the Loewe score. Every sample is a unique triplet (i, j, c) of two drugs i, j and a cell line c. Each sample has a corresponding Loewe score



(a) Transformer Encoder Architecture



(b) Encoder Architecture

Figure 7: A high-level overview of proposed changes in Transformer Encoder architecture. (a) shows original encoder from Vaswani et al. (2017). In (b) we remove positional encoder, and we replaced the self-attention mechanism with computed queries from both drugs and cell line features.

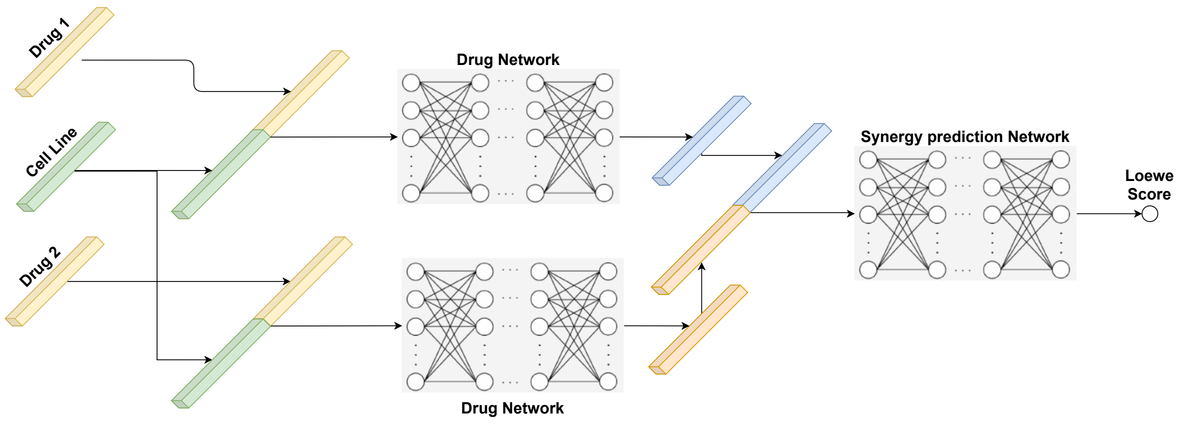


Figure 8: MatchMaker Architecture

$y_{i,j,c}$.

Cell line $c \in \mathbb{R}^{972}$ is represented by gene expression profile (Iorio et al., 2016). For the drug representation we follow Yang et al. (2019) recommendation to use hybrid representation which consists of the graph embeddings concatenated with the global molecular descriptors obtained from RDKit. To predict synergy score we use MatchMaker (Kuru et al., 2020) architecture.

MatchMaker As has been shown in the review of related work MatchMaker (Kuru et al., 2020) produces the best predictive performance on the synergy prediction task. Therefore, in this thesis we will use a model, which is constructed on the same principles that were important for the success of MatchMaker. The model has two submodels: (1) Drug specific Subnetwork (DSN) and (2) Synergy Prediction Subnetwork (SPN). Visualisation of the model can be observed in figure 8.

Drug specific Subnetwork (DSN) The DSN takes as input concatenated features of a drug and a cell line. In original implementation Kuru et al. (2020) uses two identical models, even though they serve the same purpose of getting cell-line-specific drug embedding. In our implementation we use the same model which is shared between both drugs. DSN model consists of N_{DSN} FFN layers, with *ReLU* activation function and a Dropout layer between each pair of FFN layers.

Synergy Prediction Subnetwork (SPN) The SPN receives concatenated outputs from DSN and predicts Loewe score. Because the order of the drugs does not matter, the concatenation order of $Drug_1$ and $Drug_2$ is randomly chosen for every sample. The SPN subnetwork has the same architecture as DSN - it consists of N_{SPN} FFN layers with *ReLU* and Dropout in between the layers.

The model is trained end-to-end and we minimize weighted mean squared error (MSE). Weighted loss was suggested by Kuru et al. (2020), this will make the model to focus more on the synergistic drugs, which have more therapeutic potential. The weights for every sample are proportional to the difference of its true Loewe score and the minimum Loewe score in the training set:

$$MSE_w = \frac{1}{N} \sum_{i,j,c \in D} w_{i,j,c}(y_{i,j,c} - \hat{y}_{i,j,c})^2 \tag{27}$$

$$w_{i,j,c} = \log(y_{i,j,c} - \min(\mathbf{y}) + \epsilon) \tag{28}$$

Section 5. Experiments

5.1. Experimental Setup

5.1.1 Dataset

For our experiments, we used the DrugComb dataset (Zagidullin et al., 2019), downloaded from `https://drugcomb.org/` (version 1.5, downloaded in June 2021). The dataset contains 21.5 million data points, including both monotherapy and combination therapy samples with different drug concentrations. In this thesis, we want to identify synergistic drug pairs, so that they could be prioritized for further *in vitro* experiments. Hence, first, we omitted the monotherapy samples. Secondly, if one triple (drug, drug, cell line) had multiple samples with different concentrations, we used a sample with the highest Loewe score for our final dataset. Lastly, we removed outliers where the Loewe score is lower than -200 and higher than 200. The Loewe score is represented as a percentage difference between real outcome and expected.

Our final dataset contains 338 380 samples, which is comprised of 81 cell lines and 3 081 different drugs. The distribution of the Loewe scores is shown in figure 9.

5.1.2 Dataset Splits

For the experiments, we have created 3 train-validation-test splits. First, split test model on unseen combinations of drug-drug-cell. This is the most common test split among all the publications about drug synergies. The other two splits are created to test the model's generalization power for the unseen drug/cell line during the training

Unseen combinations For the first split we ensured that all the drugs and cell lines in the test and validation sets are also present in the training set. Train, validation, and test split contain 237 700, 50 374 and 50 306 samples respectively.

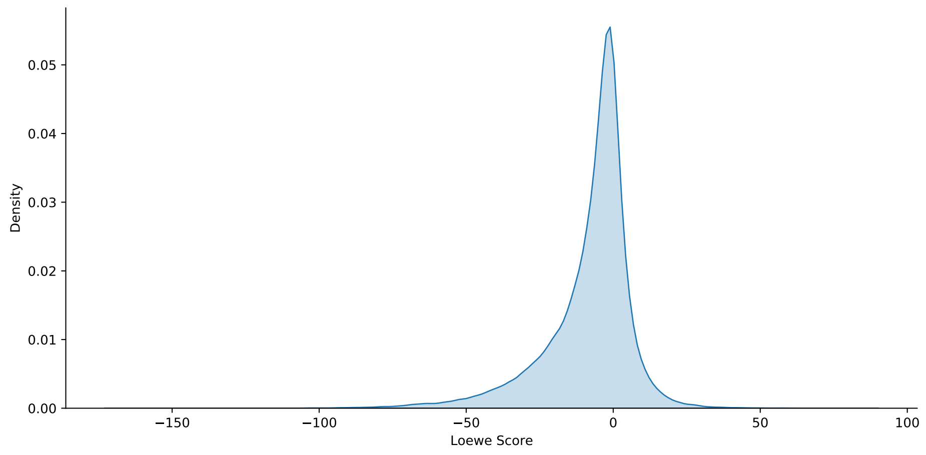


Figure 9: Distribution of the Loewe score in the processed DrugComb dataset. The dataset contains 338,380 triplets: drug1, drug2, cell line.

Unseen Cell Lines To test the generalization power of the models on unseen cell lines, for validation we selected all the samples with 11 randomly selected cell lines. Also, we removed all the samples if one of the drugs was not present in the training set. The same was done for the test set with different cell lines. 3 different splits were created with different seeds. More information about split sizes is available in appendix C.

Unseen Drugs For the validation set we selected all samples with 200 randomly selected drugs. If samples contained cell lines or a second drug that is not present in the train set, we removed these samples from the validation set. The same was done for the test set with different 200 drugs. 3 different splits were created with different seeds. More information about the split sizes in appendix C.

5.1.3 Preprocessing

DrugComb dataset consists of drug and cell line names. Obtaining drug and cell line features requires additional steps.

Molecule Features We used drug names and PubChem API (Kim et al., 2018) to extracted isomeric SMILES representation for every molecule. Afterwards, these SMILES representation were used to extract RDKit descriptors (required for hybrid representation) and all the atom and bond features, listed in tables 1, 2, with the RDKit library.

Cell Line Gene Expressions For the cell line representation, we used their untreated cells' gene expression profiles. These features for the DrugComb dataset were already computed by Kuru et al. (2020). To calculate these features gene expression profiles (Iorio et al., 2016) were obtained from the ArrayExpress database (Athar et al., 2019) (accession number: E-MTAB-3610). And the cell line is represented by gene expression profiles of the landmark genes (Subramanian et al., 2017) computed with the Robust Multi-array Average (Irizarry et al., 2003), which was downloaded from `https://www.cancerrxgene.org/`. Before training and test, we normalize cell line $c \in \mathbb{R}^{972}$ using mean and standard deviation computed using the training set.

5.1.4 Hyperparameter Optimization

As was discussed by Yang et al. (2019), hyperparameters have a great impact on the final performance of the model. To find optimal hyperparameters for the models we use Bayesian optimization using Optuna (Akiba et al., 2019) Python package. A list of the optimized parameters and their search space are shown in the appendix A. For every model, we perform 20 trials with an activated pruning option, which stops the unpromising trials early.

5.1.5 Models

Models were implemented using PyTorch framework (Paszke et al., 2019) and code is available at `https://github.com/esidorovics/dmpnn-matchmaker`. As optimizer we tried to use Adam (Kingma and Ba, 2014) and RAdam (Liu et al., 2019). The latter showed slightly better results, hence for all the experiments we used RAdam to optimize the parameters. Hyperparameters were optimized only for the unseen combinations dataset, for the other splits we used the same parameters. All the hyperparameters are available in appendix B. Models were trained on a single NVIDIA TitanRTX GPU with an early stopping on the validation MSE error and a patience parameter of 100. The average training time for

MM-DMPNN model was 6 days, for the models with multi-head attention and transformers it was 10 days.

The dataset, all the preprocessed features, and set splits are available for download[1].

As a baseline, we used MatchMaker model. Kuru et al. (2020) made available the implementation of MatchMaker, however for our experiments we re-implemented using PyTorch framework. We tested our implementation with the original implementation and the results were very similar. We attribute small deviation of the results to the different random seeds. Our implementation of MatchMaker model is available `https://github.com/esidorovics/matchmaker-pytorch`. Reasons and the implementation details are available in the appendix D.

5.2. Results

5.2.1 Unseen Combinations

Table 3 summarizes the results on the unseen combinations test sets. Models were trained on the same data splits, with 3 different seeds. Apart from models, which were introduced in the section 4, we trained two additional models: the MatchMaker model with the most frequently used molecule representation features Morgan Fingerprint and the DMPNN model with ChemoPy features (instead of RDKit library feature's). MatchMaker performs marginally better with global ChemoPy descriptors than with Morgan Fingerprint. Similarly, the MatchMaker on average performs slightly better than the DMPNN model. Because ChemoPy features proved to work well with MatchMaker, we also tried to train a DMPNN model with these features in the hybrid setting. The first run (DMPNN-Chemopy) showed that it performs significantly worse than Graph+RDKit, hence due to high computation costs we did not train more models with different seeds. DMPNN-MHA showed on average the best results across all 3 metrics, however, these improvements are not significant.

Further, we explored how model performance is improved by using a model ensemble. We averaged predictions across all 3 seeds and showed the results in the table 4. The results for all models were improved. The biggest improvements can be observed for the DMPNN-MHA model and it shows better results in comparison with MatchMaker for Pearson correlation (by 0.007) and Spearman correlation (by 0.013).

Table 3: Results on unseen combinations dataset split averaged across 3 seeds (standard deviation shown as ±). Results shown without standard deviation were trained only once.

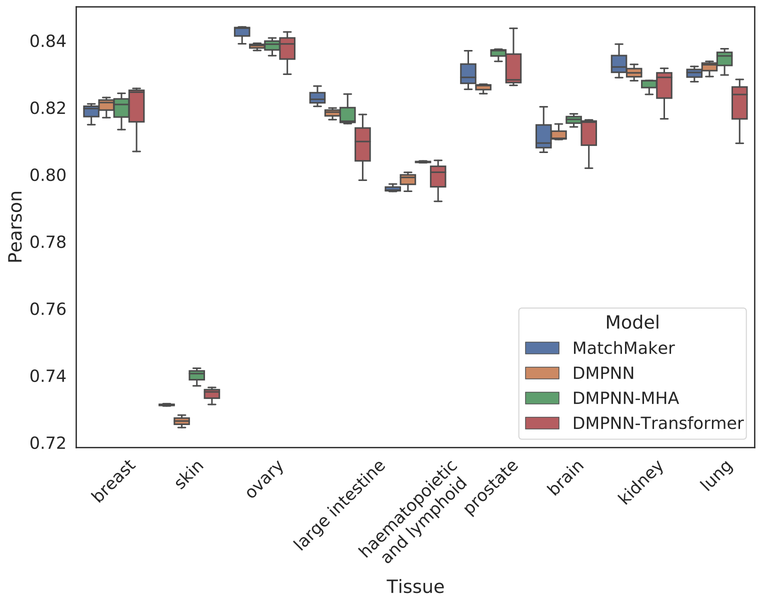
Model	Drug Features	MSE ↓	Pearson ↑	Spearman ↑
MatchMaker	ChemoPy	77.801 (±0.828)	0.803 (±0.002)	0.764 (±0.004)
MatchMaker-Morgan	Morgan Fing.	80.833 (±0.953)	0.794 (±0.003)	0.756 (±0.005)
DMPNN	Graph and RDKit	78.389 (±0.352)	0.801 (±0.001)	0.762 (±0.002)
DMPNN-Chemopy	Graph and ChemoPy	81.865	0.791	0.752
DMPNN-MHA	Graph and RDKit	**76.872** (±0.311)	**0.805** (±0.001)	**0.769** (±0.000)
DMPNN-Transformer	Graph and RDKit	77.805 (±0.729)	0.803 (±0.002)	0.766 (±0.001)

Similarly to Liu and Xie (2021) we explored tissue-specific prediction performance across the models. Figure 10 shows Pearson correlation across different tissues. Our test dataset contained also cell lines from the soft tissues, however, we excluded them due to the small sample size (60 samples). All of the models perform similarly across all the tissues and results are consistent both for Pearson and Spearman (appendix F) correlations. The two most problematic tissues are bone and skin. Low performance on

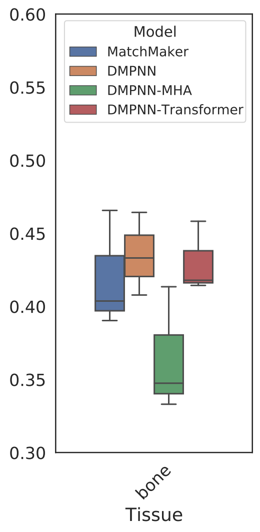
[1] `https://drive.google.com/drive/folders/1C4C4KY0ypLNzSaRLU6_0a84QONp7gPWz?usp=sharing`

Table 4: Results on unseen combinations dataset split using model ensemble. Predictions were averaged across 3 seeds.

Model	**Drug Features**	**MSE ↓**	**Pearson ↑**	**Spearman ↑**
MatchMaker	ChemoPy	73.870	0.814	0.777
DMPNN	Graph and RDKit	72.731	0.817	0.782
DMPNN-MHA	Graph and RDKit	**71.269**	**0.821**	**0.790**
DMPNN-Transformer	Graph and RDKit	73.945	0.813	0.777



(a) Tissue-specific Pearson correlation for most of the tissues



(b) Tissue-specific Pearson correlation for Bone tissues

Figure 10: Tissue-specific Pearson correlation. Bone tissues are separated to the different graph to improve readability

the bone tissue samples can be explained by the low number of samples. There are only 1 586 samples with bone tissue cell lines in the training set. While in contrast, for other tissues there are 56 001 for skin, 28 208 for kidney, 27 668 for the large intestine, 19 198 for the brain, 23 899 for ovary, 7 646 for prostate, 30 619 for lung, 23 441 for breast and 19 374 for haematopoietic and lyphoids. Interesting to note, that the same argument cannot be used to explain why all the models are performing significantly worse on the skin tissue relative to other tissues.

5.2.2 Unseen Cell Lines and Drugs

Tables 5 and 6 present results on the more challenging datasets, where every sample in the test set contains before unseen cell lines or one of the drugs. We used a simple baseline, which is the average of the other two elements seen in the training set, e.g. test sample consists of a triplet ($Drug_1$, $Drug_2$, $CellLine$) and $Drug_2$ never appeared in the training set, then for baseline, the prediction is the average of all the samples in training set where $Drug_1$ and $CellLine$ appeared together. The baseline for the unseen cell lines was computed in the same way.

DMPNN-Transformer was consistently outperforming other deep learning methods across all the metrics. However, these results are not significant and worse than the simple baseline model. Results on the unseen cell lines are significantly better than corresponding results on the unseen drug dataset.

Table 5: Results on unseen cell line dataset splits averaged across 3 different splits (standard deviation shown as ±). The baseline model is the average synergy score for 2 drug combinations in all other cell lines seen in the training set.

Model	**Drug Features**	**MSE ↓**	**Pearson ↑**	**Spearman ↑**
Baseline	-	**143.917** (±19.096)	**0.607** (±0.029)	**0.570** (±0.028)
MatchMaker	ChemoPy	156.455 (±15.340)	0.548 (±0.052)	0.494 (±0.080)
DMPNN	Graph and RDKit	155.601 (±16.262)	0.566 (±0.027)	0.530 (±0.022)
DMPNN-MHA	Graph and RDKit	158.497 (±18.093)	0.561 (±0.016)	0.526 (±0.019)
DMPNN-Transformer	Graph and RDKit	**153.730** (±17.730)	**0.574** (±0.030)	**0.537** (±0.031)

Table 6: Results on unseen drug dataset splits averaged across 3 different splits (standard deviation shown as ±). Baseline model is average synergy score for a seen drug and cell line combination in the training set.

Model	**Drug Features**	**MSE ↓**	**Pearson ↑**	**Spearman ↑**
Baseline	-	**185.007** (±38.141)	**0.383** (±0.034)	**0.365** (±0.047)
MatchMaker	ChemoPy	223.200 (±49.406)	0.265 (±0.095)	0.282 (±0.079)
DMPNN	Graph and RDKit	212.646 (±41.216)	0.304 (±0.061)	0.289 (±0.096)
DMPNN-MHA	Graph and RDKit	215.562 (±47.057)	0.298 (±0.078)	0.297 (±0.085)
DMPNN-Transformer	Graph and RDKit	**211.884** (±54.919)	**0.325** (±0.113)	**0.315** (±0.105)

5.2.3 Ablation Study

Yang et al. (2019) notes that DMPNN model does not always benefit from the hybrid representation, hence we have trained models with different molecule representations: (1) hybrid, (2) only pre-computed RDKit features and (3) only graph representation. The summary of the results is shown in table 7, due to high computation costs we have not trained models (2) and (3) with different seeds. DMPNN model benefits from the hybrid representation and performs better across all the metrics.

Table 7: Ablation study for the DMPNN model. Results on the unseen combinations dataset split, the model with hybrid representation is averaged across 3 different splits (standard deviation shown as ±)

Model	**Drug Features**	**MSE ↓**	**Pearson ↑**	**Spearman ↑**
DMPNN	Graph and RDKit	**78.389** (±0.352)	**0.801** (±0.001)	**0.762** (±0.002)
DMPNN - Only features	RDKit	81.654	0.792	0.757
DMPNN - Only Graph	Graph	81.737	0.794	0.757

In section 4 we discuss that for queries in the DMPNN-MHA model we use both drugs descriptors and gene expression data. Further, we explore if DMPNN-MHA benefits from it and compare it to the model where queries are extracted only from the RDKit descriptors for the same drug. Table 8 summarizes the results. It can be seen that the DMPNN-MHA model benefits from additional information about the other drug and the cell line and this model outperforms on all of the metrics.

Table 8: Ablation study for the DMPNN-MHA model. Results on unseen combinations dataset split, the model with computer queries from Drug 1, Drug 2 and Cell line features is averaged across 3 different splits (standard deviation shown as ±)

Model	Drug Features	MSE ↓	Pearson ↑	Spearman ↑
DMPNN-MHA	RDKit, Graph and Drug 1, Drug 2, Cell line features	**76.872** (±0.311)	**0.805** (±0.001)	**0.769** (±0.000)
DMPNN-MHA	Graph and RDKit and only Drug 1 features	83.884	0.785	0.745

5.2.4 Novel drug pair predictions

Not all possible drug-drug-cell line combinations are available in the DrugComb dataset. Even with our subset of 3081 drugs are 80 cell lines over 300 million different combinations are possible. We want to explore the most promising drug combinations, which could be tested *in vitro*. To reduce the error of our predictions we excluded cell lines from the bone, skin and soft tissues. Since the unexplored space is too big, we randomly sampled 1 million samples, which are not present in the DrugComb, and use the DMPNN-MHA ensemble of 3 models to predict the score of these combinations. In appendix G we show 50 combinations with the highest Loewe score.

5.3. Discussions

5.3.1 Molecule representation for synergy prediction

From the results above, we can see that synergy prediction models benefit from both local and global drug features. Models which were trained only on local features (MatchMaker with Morgan or DMPNN only with graph embeddings) similarly to the models, which were trained only on the global features (DMPNN model with only RDKit features) are underperforming in comparison to models which used both features: Chemopy or Graph+RDKit.

However, Liu and Xie (2021) tried to add both global and local features (physicochemical properties, toxicophores, fingerprints or graph neural networks) to their embeddings, which were based on drug-target and protein-protein interaction data, and it only had a negative effect on the synergy prediction score. We believe that there could be several reasons for that:

1. Dataset was too small. As Yang et al. (2019) pointed out it is necessary to have more than 1000 molecules to reliably use graph neural networks.
2. Embeddings based on Drug-target and protein-protein interaction may already encapsulate both global and local features, hence adding more information does not benefit the model performance.
3. With such a small dataset and so many features the model might overfit quickly.

Furthermore, we tried to visualize DMPNN embeddings and look if any pattern could be observed. Figure 11 shows the visualization of the embeddings projected into the first two t-SNE components. The visualization shows that it forms multiple clusters. For simplicity we focus mainly on the 3 biggest ones (by the number of samples) and analyzed the reasons what could cause them. We will refer to clusters as they are shown in the figure, the full list of drugs in each of these clusters is available in appendix E. Figure 12 shows the distribution of the predicted and true Loewe score for drugs within

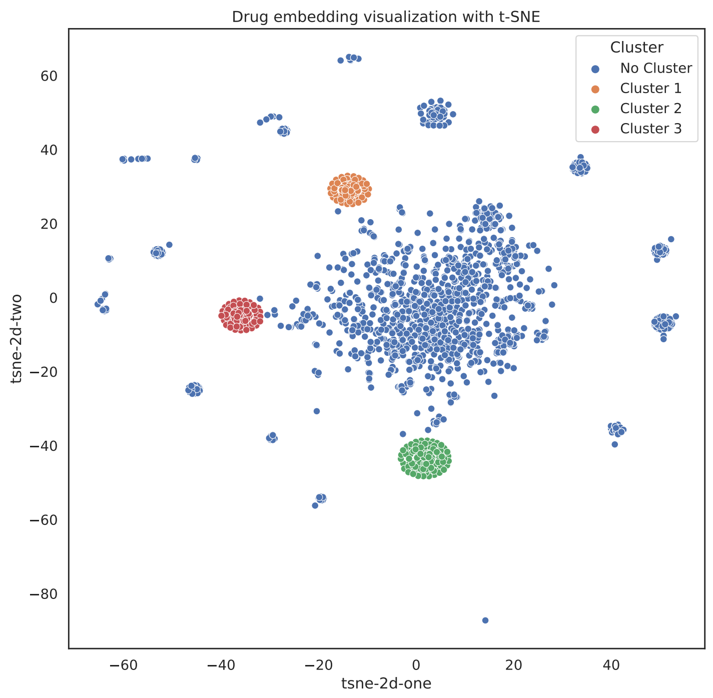


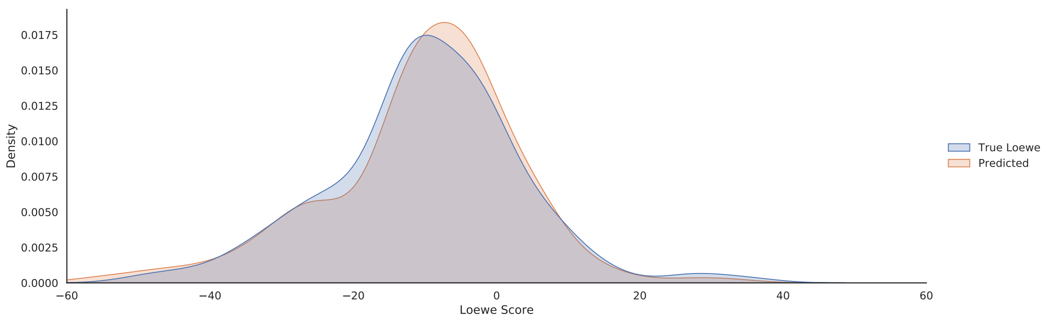
Figure 11: Visualization of DMPNN drug embeddings projected into 2D space with the first two t-SNE components

the same cluster. Such samples were quite frequent in the training set. On contrary, there are no samples in the dataset where 2 drugs are from the different clusters, therefore we do not have true Loewe scores. To obtain figure 13 we randomly sampled 20 000 triplets (drug1, drug2, cell line) from different clusters and predicted their Loewe scores. Lastly, we plotted predicted Loewe scores for every cluster interacting with other drugs outside the clusters 1-3 (figure 14).

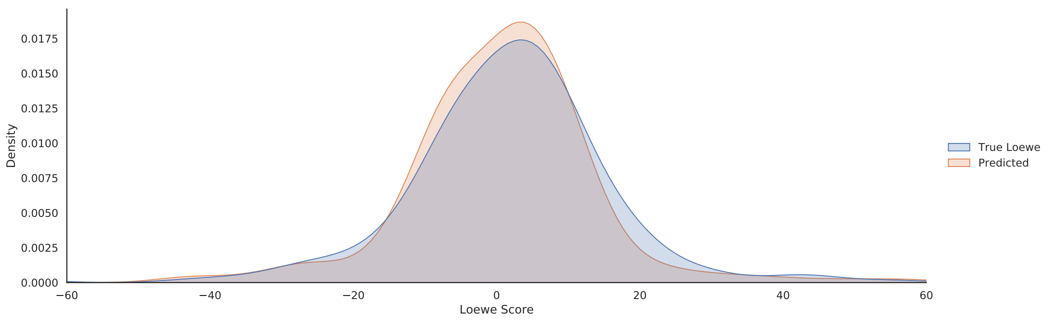
We do not see any clear pattern here. Also exploring the graphs and chemical properties did not give any insights why drugs would be clustered like that. Arguably only cluster 1 is mostly antagonistic to other drugs outside clusters, in all other graphs distributions look similar. We believe that the main cause of these clusters is overfitting. Since the only property which we could find that unites these 3 clusters is that they have interaction samples within each cluster in the training set, and these clusters never interact in the training set.

We have investigated which embeddings are better for which drugs. We compared the MatchMaker model (ChemoPy descriptors) with our best performing model DMPNN-MHA (Graph + RDKit). For every model, we have calculated MSE per drug and selected the drugs which have MSE less than 30. Table 9 shows the list of the molecules which had low MSE across all 3 seeds and were not present in the other model.

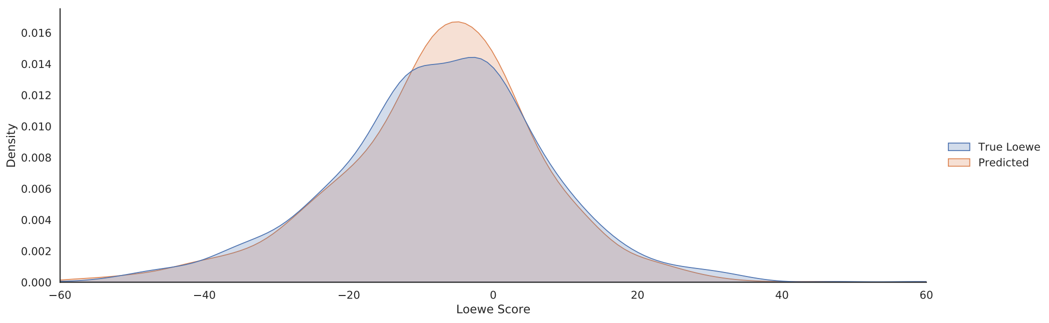
Unfortunately, we could not recognize any pattern here. Both groups have drugs with the same target, chemotherapy drugs, we also have checked the size of the molecule (by atom size) and could not see any signs of correlation.



(a) Predicted and True Loewe scores of the drugs from Cluster 1



(b) Predicted and True Loewe scores of the drugs from Cluster 1



(c) Predicted and True Loewe scores of the drugs from Cluster 3

Figure 12: Distribution of predicted and true Loewe scores for interaction between drugs withing the same cluster.

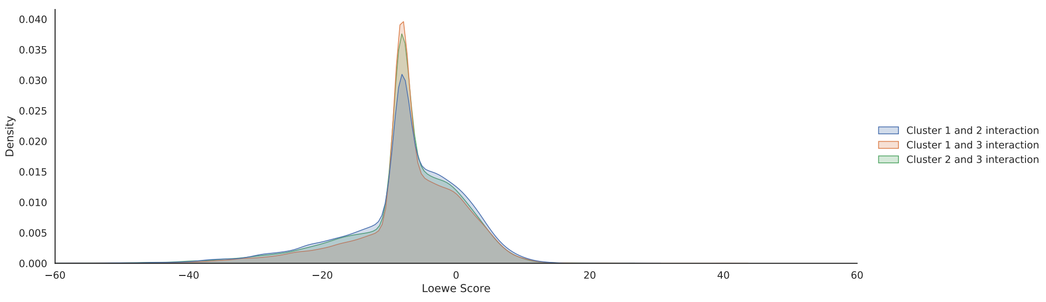


Figure 13: Predicted Loewe scores between clusters

5.3.2 Graph aggregations functions

The main difference between DMPNN and both DMPNN-MHA, DMPNN-Transfomer models is that the latter two models use global information about both drugs and cell line features to obtain the

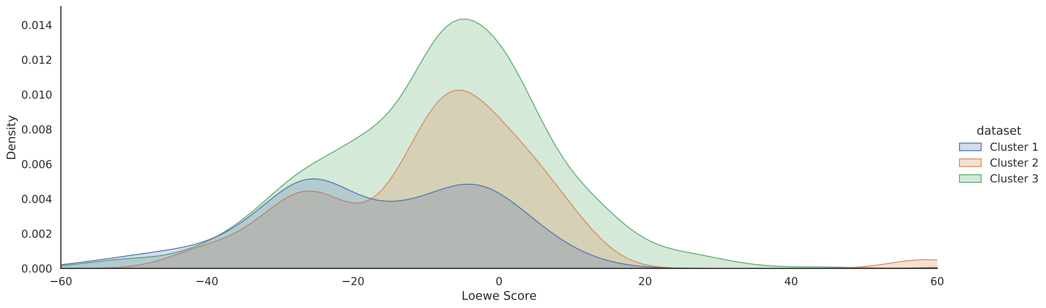


Figure 14: Predicted Loewe scores of 3 clusters with the drugs outside the selected clusters

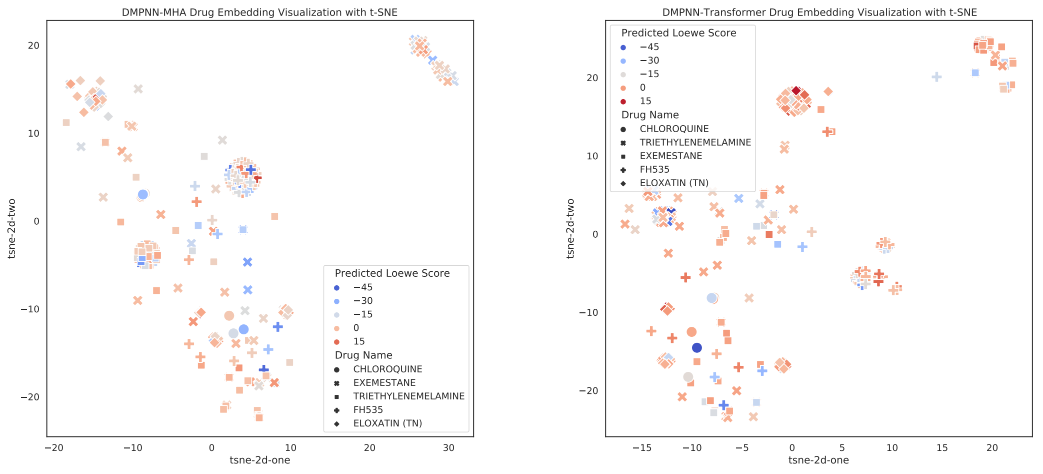
Table 9: List of molecules which are consistently better represented by MatchMaker with ChemoPy descriptors or by DMPNN-MHA with Graph + RDKit descriptors.

Model	Features	Molecule Name
MatchMaker	ChemoPy	288150-92-5, 504433-23-2, 6-AMINOHEXANOIC ACID, 944261-79-4, 955-24-8, ADAPALENE, AT7519, BUPARLISIB, CHEMBL3191144, CHIR-98014, CHIR-99021, CHLOROXINE, CLINDAMYCIN PALMITATE HCL, DANUSERTIB, FAMCICLOVIR, GYKI-52466, INDIPLON, ISODONOL, CITCO, LOTEPREDNOL ETABONATE, MELATONIN, METARRESTIN, MK-2048, NCGC00161927-01, NCGC00166236-01, NCGC00263944-01, NIMESULIDE, OMIPALISIB, PIRARUBICIN, SEMAGACESTAT, SPECTRUM 001978, SULFADOXINE, TAK-285, TAMIBAROTENE, TOXOFLAVIN, VENETOCLAX, VORICONAZOLE
DMPNN-MHA	Graph and RDKit	(E/Z)-FERULIC ACID, 185039-99-0, 356559-20-1, 371942-69-7, 6-THIOGUANINE, 722544-51-6, 951151-97-6, ALISERTIB, AMINOGLUTETHIMIDE, BALSALAZIDE, CARBOPLATIN PACLITAXEL, CHEMBL1270810, CHEMBL1950303, CHEMBL2325704, CLOPIDOGREL, COBIMETINIB (R-ENANTIOMER), ELOXATIN (TN), ERDOSTEINE, GEFITINIB, IDH-C227, MG-115, MK-2461, MOCRAVIMOD, NCGC00017354-06, NCGC00263092-01, PF-04691502, PF-3758309, PI-103, PYRONARIDINE, SINOMENINE, TELMISARTAN, TPCK, WHI-P97

drug embedding. Hence drug embedding is obtained in the context of the triplet and it may contain information about the potential synergistic effect. In figure 15 we visualized 5 randomly selected drugs and used colors to identify the Loewe score. We were expecting that these points would cluster by the synergy score, however, both plots show that synergistic and antagonistic drugs do not cluster together, which could indicate lack of generalization. Also, all drugs apart from Chloroquine form a visible cluster. The reason for that could be the number of samples for these drugs. Chloroquine is the least represented drug in the visualized example (15 samples), other drugs have substantially more samples 4751, 2994, 84, 94 for Exemesstane, Triethylenemelamine, Exolotine and FH535 respectively.

5.3.3 Generalization on Unseen Cell Lines and Drugs

Our results indicate how important the choice of a baseline is. Even though our methods do outperform MatchMaker, they still fail to generalize well to other cell lines or drugs. These results are consistent with the existing literature. Preuer et al. (2018) mentioned that DeepSynergy does fail to extrapolate for unseen cell lines and drugs, and did not even report the results.



(a) DMPNN-MHA drug embedding visualization with t-SNE.

(b) DMPNN-Transformer drug embedding visualization with t-SNE.

Figure 15: Visualization of drug embeddings of DMPNN-MHA and DMPNN-Transformer models with t-SNE.

Later Liu and Xie (2021) showed that TranSynergy generalizes better than DeepSynergy. Our reported results for unseen cell lines have similar Pearson and Spearman correlations. However, their standard deviation is smaller, which could be caused by a higher number of trainings (they used 5 splits for summarizing the results). Our Baseline is also outperforming their results.

Our results on unseen drugs are significantly worse in comparison with reported results by Liu and Xie (2021). However, we believe that the reason for such difference is much higher variety in our dataset. They used a dataset introduced by O'Neil et al. (2016), where they screened 22 experimental drugs, however the DrugComb dataset contains drugs also for chemotherapy and molecules which are not related to cancer drugs such as Vitamin D3. Even though Liu and Xie (2021) used clustering algorithms to create a more uncorrelated test set, a much lower standard deviation indicates that these drugs are still more similar than the molecules we used in our dataset. The main reason that makes us think that our results are not comparable is that Liu and Xie (2021) reported high Pearson and Spearman correlations for the DeepSynergy model. However, MatchMaker and DeepSynergy are relatively similar models, which makes it unlikely that results deviate that much. Hence we believe that on the same dataset these two models would behave similarly.

Preuer et al. (2018) hypothesized that such poor generalization is due to the lack of data. However, we think that the main reason is because we optimize our models on a simpler task, but then use the same models in the tasks where more generalization is required. All the synergy prediction models try to create models to improve the score on the unseen combination. However, it is beneficial for these models to overfit on these drugs in the training set, since the response of a random drug A will not change that much if we use it with a different molecule (this is also a drawback of the dataset we use, since most of the molecules are non-interacting). Hence if we want synergy prediction models to generalize well on unseen cell lines or drugs we should be able to sacrifice the performance on unseen combinations, or should be ready to use different models.

5.3.4 Future Work

We see two main paths which can be taken in order to build up on top of this work and improve synergy prediction. The first is to improve the input representation and the second is to work on the model which predicts the score from the inputs. Here we want to discuss few approaches which could improve the representation of the input data that should lead to a better synergistic score prediction.

As we have discussed before, worse results for the bone tissue can be due to the lack of data, however, skin performance is a symptom of bad representation of the cell lines. Liu and Xie (2021) showed that including cell dependency data together with gene expression data can significantly improve the results. In this thesis we have not tried to do it due to the time constraint, however, we believe it could be one of the fastest ways to potentially improve the results.

A recent trend in machine learning is the self-supervised learning. Some research has shown to improve results on the molecular representation (Hu et al., 2019; Park et al., 2019; Xie et al., 2021; You et al., 2020). You et al. (2020) proposed a graph contrastive learning framework for learning unsupervised representations of graph data. The authors proposed multiple graph augmentations for the model pretraining which would improve final results.

Finally, we would recommend exploring different ways to utilize the DrugComb dataset more efficiently. We have excluded information about monotherapy and ignored data about different synergistic scores with different drug concentrations. One of the ways to use this data would be to try to pretrain the network using this information and afterwards fine-tune it with the synergy dataset.

Section 6. Conclusions

In this thesis, we explored the potential of using DMPNN to learn drug embeddings for the drug synergy prediction problem. Our results on unseen drug combinations show that graph neural networks do perform on average better, however, this difference is not statistically significant. Moreover, it is not sufficient just to rely only on the learned embeddings, giving the model some insights by providing the molecular global features (hybrid representation (Yang et al., 2019)) improves the results for the synergy score prediction. Visualizing the embedding of DMPNN shows molecules form multiple clusters, however further analysis has not indicated any molecular property which could lead to such clusterization. The only property which we could find consistent among the main 3 clusters was lack of samples with the interaction between the clusters, which is a sign of overfitting.

The results also indicate the benefit of using multi-head attention architecture for aggregating node features to get the final graph representation. Even though the results are not significant, we can see significant improvement if we use models ensemble. Visualizing embeddings did not provide any intuition on why it could improve the results.

Further, we explored how graph based models perform on unseen drug and cell line dataset splits. The best performing model was DMPNN-Transformer, which on average was performing better than other deep learning approaches. However, these results are practically not usable, since even a simple baseline of predicting average performs better.

References

Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. 2015. TensorFlow: Large-scale machine learning on heterogeneous systems. Software available from tensorflow.org.

Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. 2019. Optuna: A next-generation hyperparameter optimization framework. In *Proceedings of the 25rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.

Awais Athar, Anja Füllgrabe, Nancy George, Haider Iqbal, Laura Huerta, Ahmed Ali, Catherine Snow, Nuno A Fonseca, Robert Petryszak, Irene Papatheodorou, et al. 2019. Arrayexpress update–from bulk to single-cell expression data. *Nucleic acids research*, 47(D1):D711–D715.

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. 2016. Layer normalization. *arXiv preprint arXiv:1607.06450*.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

Mukesh Bansal, Jichen Yang, Charles Karan, Michael P Menden, James C Costello, Hao Tang, Guanghua Xiao, Yajuan Li, Jeffrey Allen, Rui Zhong, et al. 2014. A community computational challenge to predict the activity of pairs of compounds. *Nature biotechnology*, 32(12):1213–1222.

Chester I Bliss. 1939. The toxicity of poisons applied jointly 1. *Annals of applied biology*, 26(3):585–615.

Leo Breiman. 2001. Random forests. *Machine learning*, 45(1):5–32.

Dong-Sheng Cao, Qing-Song Xu, Qian-Nan Hu, and Yi-Zeng Liang. 2013. Chemopy: freely available python package for computational biology and chemoinformatics. *Bioinformatics*, 29(8):1092–1094.

Alice Capecchi, Daniel Probst, and Jean-Louis Reymond. 2020. One molecular fingerprint to rule them all: drugs, biomolecules, and the metabolome. *Journal of Cheminformatics*, 12(1):1–15.

Lei Chen, Bi-Qing Li, Ming-Yue Zheng, Jian Zhang, Kai-Yan Feng, and Yu-Dong Cai. 2013. Prediction of effective drug combinations by chemical interaction, protein interaction and target enrichment of kegg pathways. *BioMed research international*, 2013.

Corinna Cortes and Vladimir Vapnik. 1995. Support-vector networks. *Machine learning*, 20(3):273–297.

Peter Csermely, Tamás Korcsmáros, Huba JM Kiss, Gábor London, and Ruth Nussinov. 2013. Structure and dynamics of molecular networks: a novel paradigm of drug discovery: a comprehensive review. *Pharmacology & therapeutics*, 138(3):333–408.

M Doherty, T Metcalfe, E Guardino, E Peters, and L Ramage. 2016. Precision medicine and oncology: an overview of the opportunities presented by next-generation sequencing and big data and the challenges posed to conventional drug development and regulatory approval pathways. *Annals of Oncology*, 27(8):1644–1646.

David Duvenaud, Dougal Maclaurin, Jorge Aguilera-Iparraguirre, Rafael Gómez-Bombarelli, Timothy Hirzel, Alán Aspuru-Guzik, and Ryan P Adams. 2015. Convolutional networks on graphs for learning molecular fingerprints. *arXiv preprint arXiv:1509.09292*.

Harry Eagle and AD Musselman. 1948. The rate of bactericidal action of penicillin in vitro as a function of its concentration, and its paradoxically reduced activity at high concentrations against certain organisms. *The Journal of experimental medicine*, 88(1):99–131.

Clement Farabet, Camille Couprie, Laurent Najman, and Yann LeCun. 2012. Learning hierarchical features for scene labeling. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1915–1929.

Evan N Feinberg, Debnil Sur, Zhenqin Wu, Brooke E Husic, Huanghao Mai, Yang Li, Saisai Sun, Jianyi Yang, Bharath Ramsundar, and Vijay S Pande. 2018. Potentialnet for molecular property prediction. *ACS central science*, 4(11):1520–1530.

Evelyn Fix and Joseph Lawson Hodges. 1989. Discriminatory analysis. nonparametric discrimination: Consistency properties. *International Statistical Review/Revue Internationale de Statistique*, 57(3):238–247.

Jerome Friedman, Trevor Hastie, and Rob Tibshirani. 2010. Regularization paths for generalized linear models via coordinate descent. *Journal of statistical software*, 33(1):1.

Jerome H Friedman. 2001. Greedy function approximation: a gradient boosting machine. *Annals of statistics*, pages 1189–1232.

Levi A Garraway, Jaap Verweij, Karla V Ballman, et al. 2013. Precision oncology: an overview. *J Clin Oncol*, 31(15):1803–5.

Kaitlyn M Gayvert, Omar Aly, James Platt, Marcus W Bosenberg, David F Stern, and Olivier Elemento. 2017. A computational approach for identifying synergistic drug combinations. *PLoS computational biology*, 13(1):e1005308.

Amin Moghaddas Gholami, Hannes Hahne, Zhixiang Wu, Florian Johann Auer, Chen Meng, Mathias Wilhelm, and Bernhard Kuster. 2013. Global proteome analysis of the nci-60 cell line panel. *Cell reports*, 4(3):609–620.

Justin Gilmer, Samuel S Schoenholz, Patrick F Riley, Oriol Vinyals, and George E Dahl. 2017. Neural message passing for quantum chemistry. In *International Conference on Machine Learning*, pages 1263–1272. PMLR.

C Pankaj Goswami, L Cheng, PS Alexander, A Singal, and L Li. 2015. A new drug combinatory effect prediction algorithm on the cancer cell based on gene expression and dose–response curve. *CPT: pharmacometrics & systems pharmacology*, 4(2):80–90.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

Liye He, Evgeny Kulesskiy, Jani Saarela, Laura Turunen, Krister Wennerberg, Tero Aittokallio, and Jing Tang. 2018. Methods for high-throughput drug combination screening and synergy scoring. *Methods in molecular biology (Clifton, NJ)*, 1711:351.

J Randolph Hecht, Edith Mitchell, Tarek Chidiac, Carroll Scroggin, Christopher Hagenstad, David Spigel, John Marshall, Allen Cohn, David McCollum, Philip Stella, et al. 2009. A randomized phase iiib trial of chemotherapy, bevacizumab, and panitumumab compared with chemotherapy and bevacizumab alone for metastatic colorectal cancer. *J Clin Oncol*, 27(5):672–680.

Sepp Hochreiter, Djork-Arne Clevert, and Klaus Obermayer. 2006. A new summarization method for affymetrix probe level data. *Bioinformatics*, 22(8):943–949.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Susan L Holbeck, Richard Camalier, James A Crowell, Jeevan Prasaad Govindharajulu, Melinda Hollingshead, Lawrence W Anderson, Eric Polley, Larry Rubinstein, Apurva Srivastava, Deborah Wilsker, et al. 2017. The national cancer institute almanac: a comprehensive screening resource for the detection of anticancer drug pairs with enhanced therapeutic activity. *Cancer research*, 77(13):3564–3576.

Genevieve Housman, Shannon Byler, Sarah Heerboth, Karolina Lapinska, Mckenna Longacre, Nicole Snyder, and Sibaji Sarkar. 2014. Drug resistance in cancer: an overview. *Cancers*, 6(3):1769–1792.

Weihua Hu, Bowen Liu, Joseph Gomes, Marinka Zitnik, Percy Liang, Vijay Pande, and Jure Leskovec. 2019. Strategies for pre-training graph neural networks. *arXiv preprint arXiv:1905.12265*.

Francesco Iorio, Theo A Knijnenburg, Daniel J Vis, Graham R Bignell, Michael P Menden, Michael Schubert, Nanne Aben, Emanuel Gonçalves, Syd Barthorpe, Howard Lightfoot, et al. 2016. A landscape of pharmacogenomic interactions in cancer. *Cell*, 166(3):740–754.

Rafael A Irizarry, Bridget Hobbs, Francois Collin, Yasmin D Beazer-Barclay, Kristen J Antonellis, Uwe Scherf, and Terence P Speed. 2003. Exploration, normalization, and summaries of high density oligonucleotide array probe level data. *Biostatistics*, 4(2):249–264.

Jia Jia, Feng Zhu, Xiaohua Ma, Zhiwei W Cao, Yixue X Li, and Yu Zong Chen. 2009. Mechanisms of drug combinations: interaction and network perspectives. *Nature reviews Drug discovery*, 8(2):111–128.

Steven Kearnes, Kevin McCloskey, Marc Berndl, Vijay Pande, and Patrick Riley. 2016. Molecular graph convolutions: moving beyond fingerprints. *Journal of computer-aided molecular design*, 30(8):595–608.

Sunghwan Kim, Paul A Thiessen, Tiejun Cheng, Bo Yu, and Evan E Bolton. 2018. An update on pug-rest: Restful interface for programmatic access to pubchem. *Nucleic acids research*, 46(W1):W563–W570.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25:1097–1105.

Halil Ibrahim Kuru, Oznur Tastan, and A Ercument Cicek. 2020. Matchmaker: A deep learning framework for drug synergy prediction. *bioRxiv*.

Greg Landrum. 2013. Rdkit: A software suite for cheminformatics, computational chemistry, and predictive modeling.

Peng Li, Chao Huang, Yingxue Fu, Jinan Wang, Ziyin Wu, Jinlong Ru, Chunli Zheng, Zihu Guo, Xuetong Chen, Wei Zhou, et al. 2015. Large-scale exploration and analysis of drug combinations. *Bioinformatics*, 31(12):2007–2016.

Liyuan Liu, Haoming Jiang, Pengcheng He, Weizhu Chen, Xiaodong Liu, Jianfeng Gao, and Jiawei Han. 2019. On the variance of the adaptive learning rate and beyond. *arXiv preprint arXiv:1908.03265*.

Qiao Liu and Lei Xie. 2021. Transynergy: Mechanism-driven interpretable deep neural network for the synergistic prediction and pathway deconvolution of drug combinations. *PLoS computational biology*, 17(2):e1008653.

S Loewe. 1928. Die quantitativen probleme der pharmakologie. *Ergebnisse der Physiologie*, 27(1):47–187.

Andrea Mauri, Viviana Consonni, Manuela Pavan, and Roberto Todeschini. 2006. Dragon software: An easy approach to molecular descriptor calculations. *Match*, 56(2):237–248.

Andreas Mayr, Günter Klambauer, Thomas Unterthiner, Marvin Steijaert, Jörg K Wegner, Hugo Ceulemans, Djork-Arné Clevert, and Sepp Hochreiter. 2018. Large-scale comparison of machine learning methods for drug target prediction on chembl. *Chemical science*, 9(24):5441–5451.

James M McFarland, Zandra V Ho, Guillaume Kugener, Joshua M Dempster, Phillip G Montgomery, Jordan G Bryan, John M Krill-Burger, Thomas M Green, Francisca Vazquez, Jesse S Boehm, et al. 2018. Improved estimation of cancer dependencies from large-scale rnai screens using model-based normalization and data integration. *Nature communications*, 9(1):1–13.

Harry L Morgan. 1965. The generation of a unique machine description for chemical structures-a technique developed at chemical abstracts service. *Journal of Chemical Documentation*, 5(2):107–113.

Melody Kay Morris, David C Clarke, Lindsey C Osimiri, and Douglas A Lauffenburger. 2016. Systematic analysis of quantitative logic model ensembles predicts drug combination effects on cell signaling networks. *CPT: pharmacometrics & systems pharmacology*, 5(10):544–553.

Vinod Nair and Geoffrey E Hinton. 2010. Rectified linear units improve restricted boltzmann machines. In *Icml*.

Michail Nikolaou, Athanasia Pavlopoulou, Alexandros G Georgakilas, and Efthymios Kyrodimos. 2018. The challenge of drug resistance in cancer treatment: a current overview. *Clinical & Experimental Metastasis*, 35(4):309–318.

Jennifer O'Neil, Yair Benita, Igor Feldman, Melissa Chenard, Brian Roberts, Yaping Liu, Jing Li, Astrid Kral, Serguei Lejnine, Andrey Loboda, et al. 2016. An unbiased oncology compound screen to identify novel combination strategies. *Molecular cancer therapeutics*, 15(6):1155–1162.

Jiwoong Park, Minsik Lee, Hyung Jin Chang, Kyuewang Lee, and Jin Young Choi. 2019. Symmetric graph convolutional autoencoder for unsupervised graph representation learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6519–6528.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc.

Kristina Preuer, Richard PI Lewis, Sepp Hochreiter, Andreas Bender, Krishna C Bulusu, and Günter Klambauer. 2018. Deepsynergy: predicting anti-cancer drug synergy with deep learning. *Bioinformatics*, 34(9):1538–1546.

William C Reinhold, Margot Sunshine, Hongfang Liu, Sudhir Varma, Kurt W Kohn, Joel Morris, James Doroshow, and Yves Pommier. 2012. Cellminer: a web-based suite of genomic and pharmacologic tools to explore transcript and drug patterns in the nci-60 cell line set. *Cancer research*, 72(14):3499–3511.

David Rogers and Mathew Hahn. 2010. Extended-connectivity fingerprints. *Journal of chemical information and modeling*, 50(5):742–754.

Kristof T Schütt, Farhad Arbabzadah, Stefan Chmiela, Klaus R Müller, and Alexandre Tkatchenko. 2017a. Quantum-chemical insights from deep tensor neural networks. *Nature communications*, 8(1):1–8.

Kristof T Schütt, Pieter-Jan Kindermans, Huziel E Sauceda, Stefan Chmiela, Alexandre Tkatchenko, and Klaus-Robert Müller. 2017b. Schnet: A continuous-filter convolutional neural network for modeling quantum interactions. *arXiv preprint arXiv:1706.08566*.

Justin S Smith, Olexandr Isayev, and Adrian E Roitberg. 2017. Ani-1: an extensible neural network potential with dft accuracy at force field computational cost. *Chemical science*, 8(4):3192–3203.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958.

Jonathan M Stokes, Kevin Yang, Kyle Swanson, Wengong Jin, Andres Cubillos-Ruiz, Nina M Donghia, Craig R MacNair, Shawn French, Lindsey A Carfrae, Zohar Bloom-Ackerman, et al. 2020. A deep learning approach to antibiotic discovery. *Cell*, 180(4):688–702.

Aravind Subramanian, Rajiv Narayan, Steven M Corsello, David D Peck, Ted E Natoli, Xiaodong Lu, Joshua Gould, John F Davis, Andrew A Tubelli, Jacob K Asiedu, et al. 2017. A next generation connectivity map: L1000 platform and the first 1,000,000 profiles. *Cell*, 171(6):1437–1452.

Ronald J Tallarida. 2011. Quantitative methods for assessing drug synergism. *Genes & cancer*, 2(11):1003–1008.

Jing Tang, Krister Wennerberg, and Tero Aittokallio. 2015. What is synergy? the saariselkä agreement revisited. *Frontiers in pharmacology*, 6:181.

Nathaniel R Twarog, Michele Connelly, and Anang A Shelat. 2020. A critical evaluation of methods to interpret drug combinations. *Scientific reports*, 10(1):1–13.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *arXiv preprint arXiv:1706.03762*.

Zhenqin Wu, Bharath Ramsundar, Evan N Feinberg, Joseph Gomes, Caleb Geniesse, Aneesh S Pappu, Karl Leswing, and Vijay Pande. 2018. Moleculenet: a benchmark for molecular machine learning. *Chemical science*, 9(2):513–530.

Fangfang Xia, Maulik Shukla, Thomas Brettin, Cristina Garcia-Cardona, Judith Cohn, Jonathan E Allen, Sergei Maslov, Susan L Holbeck, James H Doroshow, Yvonne A Evrard, et al. 2018. Predicting tumor cell line response to drug pairs with deep learning. *BMC bioinformatics*, 19(18):71–79.

Yaochen Xie, Zhao Xu, Jingtun Zhang, Zhengyang Wang, and Shuiwang Ji. 2021. Self-supervised learning of graph neural networks: A unified review. *arXiv preprint arXiv:2102.10757*.

Kevin Yang, Kyle Swanson, Wengong Jin, Connor Coley, Philipp Eiden, Hua Gao, Angel Guzman-Perez, Timothy Hopper, Brian Kelley, Miriam Mathea, et al. 2019. Analyzing learned molecular representations for property prediction. *Journal of chemical information and modeling*, 59(8):3370–3388.

Pamela J Yeh, Matthew J Hegreness, Aviva Presser Aiden, and Roy Kishony. 2009. Drug interactions and the evolution of antibiotic resistance. *Nature Reviews Microbiology*, 7(6):460–466.

Yuning You, Tianlong Chen, Yongduo Sui, Ting Chen, Zhangyang Wang, and Yang Shen. 2020. Graph contrastive learning with augmentations. *Advances in Neural Information Processing Systems*, 33:5812–5823.

Bulat Zagidullin, Jehad Aldahdooh, Shuyu Zheng, Wenyu Wang, Yinyin Wang, Joseph Saad, Alina Malyutina, Mohieddin Jafari, Ziaurrehman Tanoli, Alberto Pessia, et al. 2019. Drugcomb: an integrative cancer drug combination data portal. *Nucleic acids research*, 47(W1):W43–W51.

Xing-Ming Zhao, Murat Iskar, Georg Zeller, Michael Kuhn, Vera Van Noort, and Peer Bork. 2011. Prediction of drug combinations by integrating molecular and pharmacological data. *PLoS computational biology*, 7(12):e1002323.

A. Hyperparameter Optimizition

Table 10 provides an overview of the search space for the Bayesian hyperparameter search. Square brackets indicate range and curly brackets indicate set of tested variables. For the hyperparameter search we used Optuna (Akiba et al., 2019) library. Models were compared based on the MSE on the validation set. Two types of early stopping were applied: (1) if MSE on the validation set was not improving for 50 epochs and (2) if model performs much worse comparing comparing to previously trained model during the respective epochs.

Table 10: Search space for Bayesian hyperparameter tuning.

Feature	**MM-DMPNN**	**MM-MHA**	**MM-Transformers**
Learning rate	[1e-5, 1e-2] used log domain to sample		
Depth	[2, 6]		
Hidden size	[300, 2600] with step 100		
MM input dropout	[0, 0.6] with step 0.05		
MM dropout	[0, 0.6] with step 0.05		
Number of heads	{2, 4, 8}		
MHA dropout		[0, 0.6] with step 0.05	
# Transformer Layers			[1, 6]
Transformer hidden size			[200, 2000] with step 100

B. Hyperparameters

To train the models we used RAdam optimizer (Liu et al., 2019) and trained with early stopping based on validation MSE error with patience of 100. For predictor part of the model we used MatchMaker model with the same parameters as in original paper by Kuru et al. (2020).

Table 11: Overview of the hyperpameters for models.

Feature	**MM-DMPNN**	**MM-MHA**	**MM-Transformers**
Learning rate	15e-5	8e-5	1e-4
Optimizer	RAdam	Radam	Radam
Depth	3	2	6
Hidden size	1700	2500	1024
Dropout	0.55	0.05	0.15
MM input dropout	0.5	0.45	0.35
MM dropout	0.25	0.1	0.45
Number of heads	-	8	2
MHA dropout	-	0.0	0.25
Transformer Layers	-	-	3
Transformer hidden size	-	-	200
DSN layers	2048-4096-2048	2048-4096-2048	2048-4096-2048
SPN layers	2048-1024-1	2048-1024-1	2048-1024-1

C. Dataset Splits

Table 12: Unseen cell line dataset splits.

Seed	Test set size	Validation set size	Test set size
1	263 266	33 394	41 695
2	259 211	51 155	28 007
3	253 358	33 480	50 483

Table 13: Unseen drug dataset splits.

Seed	Test set size	Validation set size	Test set size
13	205 736	57 204	59 600
42	235 013	49 996	44 434
44	201 310	60 985	59 034

D. MatchMaker Implementation

Original MatchMaker implementation was using Tensorflow (Abadi et al., 2015) framework. However, available implementation was using resources inefficiently and since our models are based on MatchMaker, we re-implemented MatchMaker with PyTorch framework (Paszke et al., 2019).

Originally MatchMaker was trained on the DrugComb dataset version 1.4, however in our experiments we used next version 1.5. After processing the dataset, number of drugs was increased from 3 040 to 3 081. Hence we could not use drug features which were made available by Kuru et al. (2020). The authors mentioned that they used *ChemoPy* library to extract descriptors, however they did not provide list of used descriptors. After manual comparison of the features we could find source of 509 features, 32 features were left unidentified. Table 14 list feature names and their dimensionality which were used to retrain the MatchMaker model.

Table 14: Descriptors and dimensionality of the descriptors used for drug representation for the MatchMaker model.

Descriptor Name	**Size**
Basak information descriptors	21
Molecular properties	6
MOE-type descriptors	60
Autocorrelation descriptors (Geary, Moran, Moreau Broto)	96
Molecular connectivity descriptors	44
Constitutional descriptors	30
E-state descriptors	245
Kappa descriptors	7

E. DMPNN Drug embedding clusters

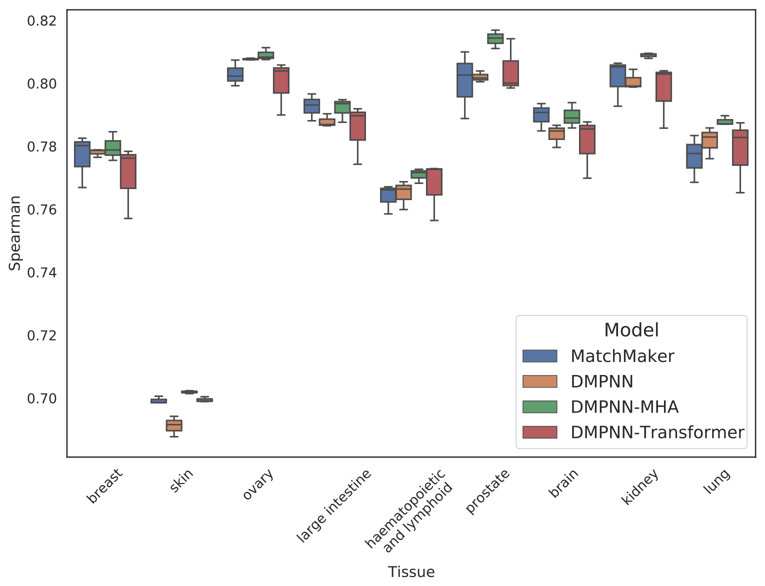
Cluster 1 CPEPCK INHIBITOR, PF-8380, GW791343, KBIOGR_000368, OXACILLIN SODIUM SALT, SR-3306, (PHENYLINDOLYL)MALEIMIDE DERIV. 79, CHEMBL2138817, SURAMIN, AZD6482, ANDARINE, LEVAMISOLE, CLOPIDOGREL, SCHEMBL326516, GW7647, 252916-29-3, CHLORAMPHENICOL PALMITATE, SECNIDAZOLE, MASOPROCOL, ML240, 3094-09-5, STAVUDINE, ZAFIRLUKAST, ALISERTIB, NCGC00090808-03, SCHEMBL6464278, PEFLOXACIN, 56990-57-9, SIPATRIGINE, 1338259-05-4, CHEMBL3348838, ISOPROTERENOL, NCGC00183677-03, LORNOXICAM, PYROXAMIDE, D-106669, ITK INHIBITOR, SUMANIROLE, ROSE BENGAL(2-), ZINC21999985, 722544-51-6, 666260-75-9, KS-00001CUZ, L-690,330, FEBUXOSTAT, LIPASE INHIBITOR, THL, CHEMBL3348935, CYCLOSPORIN A, BILOBALIDE, GM 1489, 57-22-7, PF-299804, SCHEMBL12175228, CHEMBL3348829, IRINOTECAN, BETAMETHASONE VALERATE, HOMOHARRINGTONINE, BIIB021, TRAZODONE, ARECOLINE, ZINC28538988, BLEOMYCIN SULFATE, DASA-58, ELTROMBOPAG, IDH-C227, RASAGILINE MESYLATE, EMETINE, KX2-391, NCGC00014650-03, CHEMBL492193, CARMOFUR, DOXYCYCLINE, ICRF 193, CX-5461, RAMATROBAN, METENKEPHALIN, 89565-68-4, ALOE-EMODIN, NCGC00185071-01, 145-13-1, NCGC00163532-01, ETHIONAMIDE, AMG-47A, TETRABENAZINE, 1062368-24-4, 1316214-52-4, DOXERCALCIFEROL, MECLINERTANT, GANDOTINIB, PRAVASTATIN, FLUTICASONE PROPIONATE, 494772-86-0, 936091-56-4, M344, CHEMBL3182641, SINOMENINE, KRP-297, TUBASTATIN A, LOTEPREDNOL ETABONATE, CHEMBL3185999, 209410-46-8, 30562-34-6, INDOLE-3-BUTYRIC ACID, ZINC139045355, CHEMBL3349012, DIPYRIDAMOLE, ML216, FASN BI, BESTATIN, CHEMBL3184303, BAM7, TUBACIN, PRAZOSIN, CHEMBL3348929, NATEGLINIDE, CHEMBL3348926, CHEMBL3348927, NCGC00242478-02, AZELASTINE, CHEMBL3348899, SCHEMBL2671137, GSK269962A, ROXADUSTAT, ZALCITABINE, SULFADIAZINE, TERAZOSIN, 80306-38-3, CHEMBL3184387, 252003-65-9, TELBIVUDINE, SB 202190, UNC0638, AMG-TIE2-1, CHEMBL34450

Cluster 2 AMIODARONE HYDROCHLORIDE, TACRINE HYDROCHLORIDE, CARVEDILOL, AMG-900, ALFUZOSIN HYDROCHLORIDE, GOLGICIDE A, 10402-53-6, DOXYCYCLINE HYDROCHLORIDE, ISOCONAZOLE NITRATE, SOTRASTAURIN, LIMICAN, GATIFLOXACIN, SULFAMETHIZOLE, TROPISETRON, HESPERIDIN, GRISEOFULVIN, CHENODEOXYCHOLIC ACID, 882257-11-6, CINACALCET HYDROCHLORIDE, CEPHALOMANNINE, EPINEPHRINE HYDROCHLORIDE, ETRAVIRINE, DIHYDROMYRICETIN, CATHEPSIN INHIBITOR 1, 667463-62-9, 30675-13-9, VALSARTAN, D(+)-GLUCOSE, AMILORIDE HYDROCHLORIDE, 331771-20-1, HMN-214, TYRPHOSTIN 23, CEP-32496, EPOTHILONE B, CLARITHROMYCIN, ADENOSINE, DACINOSTAT, TACROLIMUS, 2-THIOURACIL, HYDROXYZINE DIHYDROCHLORIDE, PROPRANOLOL HYDROCHLORIDE, JTC-801, METHOCARBAMOL, JSH-23, IMATINIB MESYLATE, TW-37, HESPERETIN, AZ191, VONOPRAZAN FUMARATE, MIRABEGRON, CIPROXIFAN, DIACEREIN, GUAIFENESIN, TINIDAZOLE, ILOPERIDONE, METOLAZONE, INDACATEROL MALEATE, QUININE HYDROCHLORIDE DIHYDRATE, DROPROPIZINE, SEROTONIN HYDROCHLORIDE, SCLAREOLIDE, GOSSYPOL ACETIC ACID, ESTRADIOL, URAPIDIL HYDROCHLORIDE, CHLOROXINE, IMIPRAMINE HYDROCHLORIDE, PIRFENIDONE, ASCORBYL PALMITATE, SGC0946, DICLOFENAC POTASSIUM, 1616632-77-9, MOGUISTEINE, SAFINAMIDE MESYLATE, PF-5274857, SGC-CBP30, AMINOPHENAZONE, 152121-47-6, VARDENAFIL HYDROCHLORIDE TRIHYDRATE, PICEATANNOL, EBASTINE, PANCURONIUM BROMIDE, 61350-00-3, KETOCONAZOLE, PENFLURIDOL, IRINOTECAN HYDROCHLORIDE TRIHYDRATE, 278779-30-9, PERGOLIDE MESYLATE, LOXAPINE SUCCINATE, MEVASTATIN, ATOVAQUONE, NEFOPAM HYDROCHLORIDE, TRICHOSTATIN A, BISACODYL, BENZBROMARONE, RIVASTIGMINE TARTRATE, OCTOPAMINE HYDROCHLORIDE, CALCITRIOL, 856243-80-6, CHEBI:45367, SULFACETAMIDE SODIUM, FLUFENAMIC ACID, LOMIBUVIR, 1370261-97-4, RS-127445, RIGOSERTIB SODIUM, MOROXYDINE HYDROCHLORIDE, LUBIPROSTONE, ARTEMETHER, CUDC-101, FERULIC ACID, MEPENZOLATE BROMIDE, SODIUM SALICYLATE, AMPRENAVIR, TETRACYCLINE HYDROCHLORIDE, CILAZAPRIL MONOHYDRATE, EQUOL, GENIPOSIDIC ACID, MYRIC-

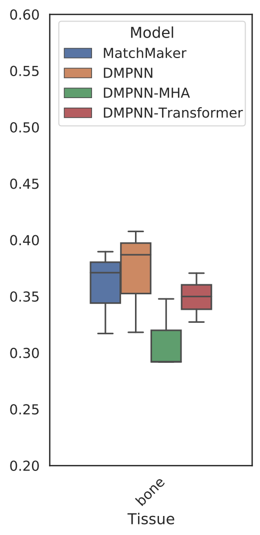
ITRIN, 155584-74-0, DYDROGESTERONE, TOLAZOLINE HYDROCHLORIDE, LUTEOLIN, SHIKIMIC ACID, BEZAFIBRATE, OXELADIN CITRATE, 1118807-13-8, CLOSANTEL, LACIDIPINE, PRANOPROFEN, PROPAFENONE HYDROCHLORIDE, CARBIMAZOLE, SPIRAMYCIN, DARIFENACIN HYDROBROMIDE, BUTENAFINE HYDROCHLORIDE, BRIMONIDINE TARTRATE, PROTHIONAMIDE, BENZTROPINE MESYLATE, RGFP966, CGS 21680 HYDROCHLORIDE, VILAZODONE HYDROCHLORIDE, CLINDAMYCIN PALMITATE HCL, AT9283, ICG-001, 55778-02-4, SULINDAC, HEXESTROL, DEXLANSOPRAZOLE, OXETHAZAINE, 280744-09-4, TIZANIDINE HYDROCHLORIDE, IRBESARTAN, 1246086-78-1, ESMOLOL HYDROCHLORIDE, U0126-ETOH, BRUCINE, GABEXATE MESYLATE, NISOLDIPINE, CANAGLIFLOZIN, AMG-517, AURORA A INHIBITOR I, FLORFENICOL, TAZEMETOSTAT, TRIMETHOPRIM, EFAPROXIRAL SODIUM, IPRATROPIUM BROMIDE, TEMSIROLIMUS, 732302-99-7, S-RUXOLITINIB, LOPERAMIDE HYDROCHLORIDE, DYNAMIN INHIBITOR I, DYNASORE, 386750-22-7, (-)-BLEBBISTATIN, LAMIVUDINE, LIDOCAINE, TIRATRICOL, RIFAPENTINE, FENBENDAZOLE, DANOPREVIR, ONALESPIB, OLSALAZINE SODIUM

Cluster 3 SECINH3, AIM-100, XARELTO, (E/Z)-FERULIC ACID, RILMENIDINE, AVN-944, ZINC84688828, MECARBINATE, DRONEDARONE, ESTRAMUSTINE PHOSPHATE SODIUM, VARENICLINE, IDELALISIB, PH-797804, GANCICLOVIR, BMS-599626, GEMFIBROZIL, BMS-265246, LOMEFLOXACIN, 372196-67-3, HEMATOXYLIN, DL-CARBIDOPA, AR-C155858, 2-DEOXY-D-GALACTOPYRANOSE, NCGC00161927-01, 1194044-20-6, CHEMBL1316023, FLUVOXAMINE, TROXATYL, 96036-03-2, ZINC11616871, FORMESTANE, RESISTOMYCIN, 504433-23-2, 207679-81-0, SULFANILAMIDE, CARDIOGENOL C, TRANS-CAFFEIC ACID, NOCODAZOLE, 94875-80-6, AMG-51, TROGLITAZONE, LUMINESPIB, NCGC00263280-01, 314042-01-8, SPECTRUM_001978, AMINOGLUTETHIMIDE, SIVELESTAT, NITROFURAZONUM, PURMORPHAMINE, TEMOCAPRIL HYDROCHLORIDE, BMS-777607, NEDAPLATIN, FESOTERODINE, MEBENDAZOLE, CHEMBL3349004, 4-METHYLPYRAZOLE, 128517-07-7, FASIGLIFAM, CHEMBL3348892, 110025-28-0, CHEMBL3348941, VENLAFAXINE, NCGC00346475-01, CDK/CRK INHIBITOR, CYTIDINE, MIANSERIN, KINETIN, ORG 25935, SEPANTRONIUM, ZINC100015731, FLEROXACIN, FLURBIPROFEN, 1,25-DIHYDROXY VITAMIN D3, 926319-75-7, WHI-P97, SEMAGACESTAT, ZILEUTON, SB-3CT, DABIGATRAN, 124936-75-0, AMIODARONE, GSK837149A, 1144068-46-1, ROFECOXIB, CARISOPRODOL, NCGC00162423-03, HYDROXYCHLOROQUINE, TRIOXSALEN, BINDARIT, CYTISINE, SA-13353, GENTIAN VIOLET CATION, GNE-477, CHEMBL3348900, CHEMBL3183697, 9-CIS ACITRETIN, SR9011, CHEMBL3348950, TM30089, ALPELISIB(BYL-719), ERGOSTA-5,7,22-TRIEN-3-OL, IDEBENONE, CHEMBL3348905, ICARITIN, 13292-46-1, 309928-48-1, 8-METHYL-N-VANILLYL-6-NONENAMIDE, BALOFLOXACIN, RABEPRAZOLE(1-), REBASTINIB, 648450-29-7, TETRANDRINE, AMOXAPINE, 301836-41-9, PEPSTATIN, COSTUNOLIDE, NUTLIN-3B, NICOTINAMIDE, L-HYOSCYAMINE, CHEMBL3348870, NINTEDANIB, TALADEGIB, BCP07985, EHT-1864, CUDC-907, PHENOXYBENZAMINE, ACHP, TRIFLUOPERAZINE, GSK 525768A, PHA-408, 500579-04-4, TGX-221, 1-BETA-D-XYLOFURANOSYLURACIL, FASCAPLYSIN, NCGC00346524-01, BETAMETHASONE DIPROPIONATE, CHEMBL3348959, HYDROXOCOBALAMIN, CVT-6883, OPREA1_748549, D-CYCLOSERINE

F. Tissue-specific Spearman correlations



(a) Tissue-specific Spearman correlation for most of the tissues



(b) Tissue-specific Spearman correlation for Bone tissues

Figure 16: Tissue-specific Spearman correlation. Bone tissues are separated to the different graph to improve readability

G. Novel drug pair predictions

Table 15: Highest predicted Loewe scores for unseen drug combinations in part of the DrugComb Dataset.

Drug combination	Cell line	Loewe Score
KPT-276 and SGI-1027	T98G	62.77
CBIOL_002052 and PIMASERTIB	T98G	56.95
RUCAPARIB PHOSPHATE and TICARCILLIN SODIUM	T98G	55.13
690206-97-4 and SNX-2112	L-1236	53.83
BUTHIONINE SULFOXIMINE and PHENOXYBENZAMINE	T98G	53.11
GSK 525768A and 83-88-5	T98G	52.12
1392399-03-9 and SULBACTAM	T98G	49.65
CHEMBL166161 and ATORVASTATIN CALCIUM	T98G	49.30
SGI-1776 and AZD7545	T98G	48.57
COUMARIN and PIMASERTIB	T98G	47.68
RESERPINE and SARACATINIB	T98G	47.42
NEVIRAPINE and AC1MDWZ9	T98G	47.36
TIZANIDINE and 1392399-03-9	T98G	46.84
PIBOSEROD and 873225-46-8	T98G	44.82
CHEMBL494027 and HMS3673C13	T98G	44.28
BLEOMYCIN and OXIBENDAZOLE	T98G	43.97
CLARITHROMYCIN and TIDEGLUSIB	T98G	43.40
DOXOFYLLINE and PAEONOL	T98G	43.28
KW-2478 and 6-MERCAPTOPURINE	T98G	43.07
L-690,330 and 1392399-03-9	T98G	42.17
PIMASERTIB and NCGC00094763-03	T98G	41.92
NCGC00185736-01 and NU6102	T98G	41.65
LAFUTIDINE and DOLUTEGRAVIR	T98G	41.50
GOLGICIDE A and KI8751	T98G	40.81
KETOROLAC and TADALAFIL	T98G	40.67
TASISULAM and CX-5461	T98G	40.61
AZD5582 and PRALATREXATE	HCT116	40.55
NIFEDIPINE and 10030-85-0	T98G	40.26
ENZASTAURIN and CANERTINIB	L-1236	39.82
ADEFOVIR DIPIVOXIL and CETRIMONIUM BROMIDE	T98G	38.49
THIAZOVIVIN and NVP-BHG712	T98G	38.42
IWP-L6 and ZINC21999985	T98G	38.24
LOSARTAN POTASSIUM and POSACONAZOLE	T98G	37.14
FASN BI and GM 1489	T98G	37.12
N-ACETYL-L-CYSTEINE and EBPC	T98G	37.10
RUXOLITINIB and NCGC00163548-02	SKMES1	36.56
SGI-1027 and GLAFENINE	T-47D	36.45
BIBR-1532 and RG2833	T98G	36.19
CBIOL 001725 and RG2833	SN12C	36.07
THEOBROMINE and 501925-31-1	T98G	35.95
873225-46-8 and PD123319	T98G	35.25
CETYLPYRIDINIUM CHLORIDE and SGI-1027	CCRF-CEM	34.68
SCH772984 and EMRICASAN	T98G	34.31
WAY-100635 MALEATE SALT and THIAMPHENICOL	T98G	34.02
PF-299804 and CHEMBL1090466	SNB-75	33.66
AZATADINE MALEATE and NVP-BHG712	T98G	33.53
CHIR-99021 HCL and PARBENDAZOLE	T98G	33.20
AC-261066 and LEVODOPA	T98G	33.08
OLAPARIB and NCGC00163548-02	L-1236	33.05
1392399-03-9 and SARACATINIB	T98G	32.94