Learning Distributed Representations of Drugs and Side Effects for Predicting Adverse Effects of Drug Combinations

Rubén E. Jiménez F.

Supervisor: Prof. Alberto Paccanaro

Thesis submitted in fulfilment of the requirements for the Degree of

Doctor of Philosophy

Department of Computer Science

School of Engineering, Physical and Mathematical Sciences

Royal Holloway, University of London

July, 2022

Declaration of Authorship

I Rubén E. Jiménez Franco hereby declare that this thesis and the work presented in it is entirely my own. Where I have consulted the work of others, this is always clearly stated.

Name of the candidate

Date

Dedicated to the loving memory of
Abuelo Guillermo

Though you never got to see this, you are in every page...

Acknowledgements

Foremost, I would like to express my gratitude to my supervisor Prof. Alberto Paccanaro. His work ethic, scientific approach, and great passion for research inspired me greatly throughout these years. I will always value the freedom to pursue my research interests and the keen guidance that he provided me during my time as his student. I am forever grateful for his advice and support, especially during the hardest times of this journey.

My research would not have been possible without the supportive funding from the College Studentship awarded to me by the Department of Computer Science at Royal Holloway, University of London.

I thank everyone at PaccanaroLab for making this ride more joyful. A special thanks to Diego, Phil, and Santiago who actively collaborated with my research through constructive discussions and insightful advice.

I owe a deep sense of gratitude to my family, for all their continued love and support. Despite the great distance separating us, I felt them accompanying me closely throughout these years. A heartfelt thanks to my parents Guillermo and Marta for their unwavering belief in me, it truly means the world to me.

I would also like to acknowledge with gratitude, the support and companionship of Gabriela. Finally, I want to thank my friends, for all the cherished moments spent together. In particular to Alex, Nata, Nery and Nick, I thank you for all the talks, laughs, banter, words of advice, venting sessions, and especially for the encouragement and support you always show me.

Contents

List of Abbreviations

DDI	Drug-Drug Interaction
ADE	Adverse Drug Event
ADME	Absortion, Distribution, Metabolism, Excretion
WHO	World Health Organisation
PV	Pharmacovigilance
FDA	Food and Drug Administration
AERS	Adverse Event Reporting System
DCSE	Drug Combinations Side Effects
DNN	Deep Neural Network
SMILES	Simplified Molecular-Input Line-Entry System
SSP	Structural Similarity Profile
ECFP	Extended-Connectivity Fingerprint
PCA	Principal Component Analysis
MLP	Multi-layer Perceptron
SIDER	SIDE Effect Resource
NMF	Non-negative Matrix Factorisation
AUROC	Area Under the Receiver Operator Characteristic Curve
AUPRC	Area Under the Precision Recall Curve
ReLU	Rectified Linear Unit
CNN	Convolutional Neural Network
SGD	Stochastic Gradient Descent
LS	Least Squares
MUR	Multiplicative Update Rule
LogNMF	Logistic Non-negative Matrix Factorisation

SCRUB	Statistical Correction of Uncharacterised Bias
NDCG	Normalised Discounted Cumulative Gain

List of Figures

List of Tables

List of Algorithms

Abstract

Drug combinations are commonly used to treat complex diseases or co-existing conditions. However, they pose a risk of side effects due to adverse Drug-Drug Interactions (DDIs). During drug development, the detection of DDIs remains an intractable problem mostly due to the relatively small sample size in clinical trials and the sheer number of possible drug combinations. Recently, due to the development of Adverse Event Reporting Systems (AERs), datasets have become available that collect data on adverse side effects caused by drugs already on the market. These resources encouraged the development of machine learning methods to predict side effects caused by pairs of drugs.

In this thesis, I present DCSE-twin (pron. Dixie-twin), the main approach I developed for predicting adverse effects of drug pairs. I also show the early models that led to the development of DCSE-twin. The main idea behind my approach is to learn distributed representations separately for drugs and side effects. These are learned in a neural network that also combines non-linearly the representations for single drugs to obtain the representations for drug pairs. A key finding from my early models is that non-linearity is crucial to model the combination of the individual drugs.

The existence of separate explicit representations for drugs, side effects and drug combinations allows my model to synergistically combine learning from side effect data of both single drugs and drug combinations. In this way, my model is able to answer new pharmacologically relevant questions, being able to predict side effects for drug pairs for which no side effects are known, or even for drug pairs in which neither of the drugs is known to interact with any other drug (these are known as "cold start" problems). DCSE-twin is tested in a variety of experimental procedures, including prospective evaluations, and its performance is compared against state-of-the-art methods. I also introduce several novel experimental settings in an attempt to approximate real-world scenarios in drug development, pharmacovigilance and drug repurposing. The results presented here show that DCSE-twin outperforms existing state-of-the-art methods on every experimental setting. Moreover, these experiments allow me to shed some light on biological aspects of the problem including the need for non-linearly combining representations for single drugs to obtain the representation for drug pairs. DCSE-twin represents an opportunity to use patient population data to flag and prioritise adverse effects caused by drug combinations for further validation.

"Science, my lad, is made up of mistakes, but they are mistakes which are useful to make, because they lead little by little to the truth."

Jules Verne

1 Introduction

Many human diseases are caused by highly complex biological processes that are resistant to the activity of a single drug [5]. A common strategy to combat these diseases, and overcome these limitations, is polypharmacy. This refers to a therapy involving the concurrent use of medications, also called drug combinations [6]. Drug combinations are found to be highly useful for enhanced therapeutics as they can involve synergistic effects that are triggered by Drug-Drug Interactions (DDIs), for example, by modulating the activity of distinct proteins [7, 8]. These DDIs can be exploited to repurpose drug combinations to obtain optimal treatment for many complex conditions, such as heart failure, severe hypertension or cancer [9]. Drug repositioning refers to the process of finding new uses outside the scope of the original medical indication for existing drugs [10]. Repurposing existing drugs in combinations not only provides new therapeutic possibilities but also reduces the cost and time for drug development [5]. For example, the combination of Venetoclax and Idasanutlin has recently been shown to lead to superior antileukemic efficacy in the treatment of acute myeloid leukaemia [11]. In this case, the two drugs improve survival by simultaneously targeting complementary mechanisms.

Although the use of polypharmacy therapies may be needed for the treatment of many diseases [12], a major drawback is the increased risk of adverse side effects that are due to the DDIs. Adverse DDIs occur when the activity of one drug may be alerted significantly due to the presence of other drugs [13, 14]. They constitute one of the most important concerns in clinical rational administration and postmarketing pharmacovigilance. Polypharmacy side effects are becoming an increasingly serious problem in health care, affecting over 15% of the U.S. population and costing >170 billion a year for treatment [1]. About 30% of all unexpected Adverse Drug Events (ADEs) may be accounted for adverse DDIs [15]. ADEs are a leading cause of morbidity and mortality around the world [16].

DDIs are classified as pharmacokinetic or pharmacodynamic, according to their underlying mechanism of action [9, 17]. Pharmacokinetic interactions are those in which one drug affects the Absorption, Distribution, Metabolism or Excretion (ADME) of another drug. These interactions can be quantified by the changes in kinetic parameters such as the area under the curve, half-life or peak serum concentration. The most common type of pharmacokinetic drug interaction involves drug metabolism, in particular, reactions with certain liver enzymes play a key role. Some drugs induce these enzymes (e.g., rifampin, phenobarbital, phenytoin, carbamazepine, efavirenz), and many more drugs are inhibitors (e.g., some macrolide antibiotics, antifungal azoles, HIV protease inhibitors, immunosuppressants, antidepressants, proton pump inhibitors, antitumoral). Naturally, the enzyme inducers cause an increase and the inhibitors a decrease in drug metabolites, with consequences that vary according to the type of metabolite; drugs can be metabolized to active, inactive or toxic forms. Pharmacodynamic interactions are those where the effects of one drug are changed by the presence of another, without alterations in drug kinetics. Sometimes these interactions are due to competition at the level of drug receptors, but often the mechanisms are indirect and involve interference with physiological mechanisms. There are additive or synergistic interactions that cause an increase in pharmacological effects, and antagonistic interactions that cause a decrease and can often lead to adverse side effects [9].

Adverse DDIs constitute an ever-growing issue, as the number of prescriptions of two or more drugs for the treatment of diseases continues to increase [4]. Also, the number of approved drugs keeps increasing, and with this, the number of potential interactions between prescribed medications rapidly rises. As the population is ageing and polypharmacy is becoming progressively more common, there is an increased likelihood of DDIs that can inadvertently lead to exaggeration of adverse effects. This kind of event cannot be properly prevented without recognising the need to adjust medications based on the potential risks brought by DDIs. Normally, health operators are unaware of the clinical risks of certain drug combinations. And the fact is that potential DDIs far outnumber the known actual drug interactions.

Unintentional and mismanaged DDIs are a common reason for preventable polypharmacy side effects. Therefore, there is a need for carefully planned preclinical and clinical DDI studies during drug development, and also during the postmarketing stage after drug approval. Methods to identify DDIs earlier in a comprehensive way can be implemented to improve clinical decision making [18]. The World Health Organisation (WHO) estimates that at least 60% of adverse drug reactions are preventable [17]. Discovering and predicting DDIs will help prevent life-threatening consequences for patients and it will prompt safe drug co-prescriptions for better treatments.

1.1 Detection and prediction of Drug-drug interactions

Pharmacovigilance (PV) is referred to as drug safety monitoring. WHO formally defines it as "the science and activities relating to the detection, assessment, understanding and prevention of drug-related problems" [19]. PV can be classified into two categories: (1) pre-authorisation PV which concerns the information that ADEs accumulated from clinical trial settings[20]; and (2) postmarketing/post-authorisation PV concerning the safety information collected during the drugs post-authorisation life cycle. Pre-authorisation PV is unable to fully detect drug safety concerns before its use in the real world. This is due to factors such as:

- the restricted sample of trial participants as compared to the total targeted population intended for the drug's market,
- the limited interval of drug exposure per each study participant. This is even

more accentuated if the drug is intended for chronic use,

- a lack of possibly risky patient subpopulations who are usually omitted from clinical trials [21].

Even considering these aspects, the premarketing information about a drug's efficacy is normally more reliable and comprehensive. However, a complete safety profile cannot be established during the premarketing stage [22].

Additionally, premarketing trials don't usually explore DDI studies, and are more focused on the safety and efficacy profiles of individual medications. Pharmacology profiling leads to the identification of both intended and unintended single drug-induced effects, such as biological system perturbations [3]. In contrast, patients on drug combinations are not usually even included in the clinical trials. Furthermore, it is unfeasible to test all possible pairs of drugs and their side effects. This means that many potentially serious polypharmacy side effects remain unobserved in the relatively small clinical testing [6, 23]. Therefore, reliable automated prediction of such risks is highly desirable to mitigate their impact on patients.

To address the issue of small sample size in clinical trials, the U.S. Food and Drug Administration (FDA), WHO, and Health Canada have created large Adverse Event Reporting Systems (AERSs) that collect data from clinicians, patients, and pharmaceutical companies. This resource represents an opportunity to monitor drug safety in a larger and more diverse population of patients [23]. The abundance of data led to the development of many signal detection algorithms that quantify the "unexpectedness" of an ADE being reported for a drug. Unpredictable DDIs, which can only be explained by the interaction between the two drugs and not by any of the individual drugs on their own, are recognised only through such signal detection practices during the postmarketing life cycle [24].

In the past decade, a collection of DDI-related information has become publicly available. The TWOSIDES dataset [23] is the result of a data-driven approach to flag potentially significant drug-drug-event associations in the AERS. It includes information about adverse side effects caused by the combinations of two drugs.

The availability of TWOSIDES generated a surge of novel computational methods that are developed for the prediction of adverse DDIs (see Chapter 2). Generally speaking, these methods can be separated into two categories, based on the specificity level of their predictions. The first category of methods focuses on predicting if a given drug pair will have an adverse DDI or not. This is framed as a binary prediction problem. If we consider the example in Figure 1.1, this type of method will try to predict the probability of Aspirin and Warfarin to having an adverse DDI. The second category of methods predicts the specific type of polypharmacy side effect associated with a given drug pair. Following the example in Figure 1.1, this type of method will provide a probability of the side effect *of gastrointestinal haemorrhage* being associated with the drug pair consisting of Aspirin and Warfarin.

In my work, I focus on the problem of predicting specific polypharmacy side effects. That is, given two drugs, d_i and d_j, and a specific side effect type r, my goal is to assign a score to the triplet (d_i, d_j, r). This score can be interpreted as the probability of the side effect r being caused by the co-administration of drugs d_i and d_j.

To this end, this thesis introduces a new approach that I have called DCSE-twin (pronounced *dixie twin*), for the prediction of adverse DDIs. DCSE stands for Drug Combinations Side Effects, and twin denotes the fact that my approach consists

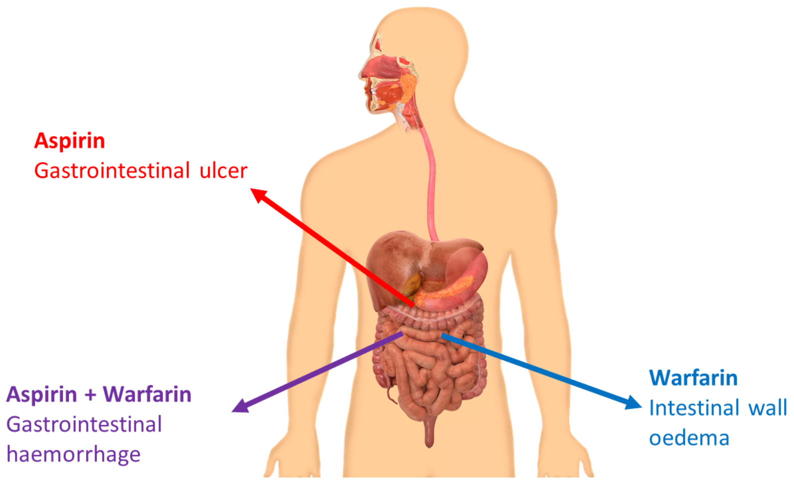


Figure 1.1: Example of a polypharmacy side effect. Aspirin on its own can cause the side effect *gastrointestinal ulcer*, and Warfarin can cause *intestinal wall oedema*[a]. However, when the two drugs are prescribed together, they have an increased risk of causing *gastrointestinal haemorrhage*, which can only be explained by the combination of the two drugs.

a: Oedema is a build-up of fluid in the body which causes the affected tissue to become swollen. https://www.nidirect.gov.uk/conditions/oedema

of two similar models that work together. DCSE-twin learns low-dimensional distributed representations for drugs and side effects, simultaneously, from side effect data of both single drugs and drug pairs. DCSE-left is a logistic non-negative matrix factorisation model that learns representations to predict off-label side effects for single drugs. The other component, DCSE-right, is a model that combines the individual distributed representations of the single drugs in a nonlinear way, to learn a representation for the pair of drugs. The main objective of my approach is to learn these distributed representations to predict a score for a given (d_i, d_j, r) triplet. The idea behind these representations is that they can encapsulate meaningful information about biological activity that is relevant for explaining and predicting adverse DDIs.

1.2 Distributed representations

The notion of distributed representations was introduced in 1984 by Geoffrey E. Hinton [25]. Each of the entities involved in the representation, is represented by a pattern of activity distributed over many computing elements, and each computing element is involved in representing many entities. This concept has then been applied with considerable success to different tasks, such as statistical language modelling, speech recognition, and a wide range of natural language processing tasks [26].

An advantage of this type of representation is that it leads to a good automatic generalisation. When the weights in the elements, that compose the representations, are changed to incorporate new knowledge about one entity, the changes affect the knowledge associated with other entities that are represented by similar activity patterns [27]. This weight adjustment results in the elements representing important features of the task domain and the regularities in the task are captured by the interaction between these units [28].

My work focuses on learning distributed representations for drugs, pairs of drugs, and side effects. The idea is to have the distributed representations capture important biological features involved in the mechanisms of ADEs. Ideally, the problem of predicting polypharmacy side effects can be modelled by the interaction between the units of the distributed representations of the drug pair and the side effect. These representations provide the ability to capture similarities, between drugs or side effects, by the similarity of their representations. Consequently, they allow the ability to generalise new information in sensible ways by learning how to represent entities with a distributed pattern of activities.

My approach, DCSE-twin, learns the distributed representations for drugs and side effects by integrating information from two different task domains, namely, off-label side effects for single drugs, and polypharmacy side effects. This allows the model to change the weights of the representations in accordance with the new knowledge being added. Specifically, when one model modifies the representations, these changes affect the knowledge associated with drugs and side effects, in the other model, which is represented by similar activity patterns, independently of the fact that they belong to different problems or models.

1.3 Document organisation

This thesis is organised as follows. Chapter 2 discusses the most relevant existing methods in the current DDI prediction literature. It includes an explanation of the baseline methods I compare my approach against and highlights the limitations in the existing literature. Chapter 3 details the preliminary models that paved the way for my final approach and explains the main findings from these methods. Chapter 4 presents my main proposed method for the prediction of adverse DDIs, DCSE-twin. It also includes the formal mathematical definitions for both DCSE-left and DCSE-right. In Chapter 5, I showcase the experimental results. I provide the comparison of my method against different baselines and the different evaluation protocols. I also provide different experiments to further understand the inner workings of DCSE-twin. Chapter 6 contains the conclusion of this thesis and a discussion about possible avenues for future work. Lastly, the Appendix A includes an alternative nonlinear combination for DCSE-right and some further experiments about the learning of DCSE-twin and the effects of regularisation.

1.4 Contributions of this thesis

In this thesis, I propose a novel approach, DCSE-twin, for the prediction of polypharmacy side effects, see Chapter 4. I thoroughly compare DCSE-twin with the state-of-the-art under several experimental settings. The experiments show that my approach consistently outperforms the competitors and obtains the best performance for predicting adverse DDIs (see Chapter 5). Other specific contributions of this thesis are listed below, along with the specific section(s) where each item can be found in the document.

- My work is able to address the *cold start* problem in which a prediction can be made for drugs, pairs of drugs or side effects without any previous interaction information (i.e. unseen during training). I propose novel experimental settings that approximate more realistic and interesting scenarios, than the most common settings used in the current literature. I also provide the biological research questions that each setting attempts to answer. I show that the performance of DCSE-twin is consistently better than the state-of-the-art, across the different evaluations, see Chapter 5.1.3, 5.1.5.

- My work is the first to introduce a prospective evaluation for adverse DDIs. This evaluation entails predicting newly discovered adverse DDIs with a model trained with older data. This is, to the best of my knowledge, the first work that introduces the chronological aspect of DDI prediction. This replicates a real-world scenario even more accurately, since the DDIs that I am predicting, were discovered after the ones used for the training of the model, see Chapter 5.1.4.

- I am the first to clearly evaluate the case of predicting true polypharmacy side effects. A drug pair can cause a side effect that can already be explained by either of the drugs individually. However, a more important and interesting case is when the drug pair causes a new side effect that can only be explained by the co-administration of the two drugs. To the best of my knowledge, this distinction was never made explicit in the current literature on DDI prediction. See Chapter 5.1.6.

- I demonstrate how the problem of learning distributed representations for pairs of drugs is highly nonlinear. I provide empirical evidence to show that using linear models is not sufficient to model the complex relationships between two drugs, see Chapters 3.3, 5.2.

- I propose a novel loss function and optimisation algorithm for learning a regularised logistic non-negative matrix factorisation. I derive the multiplicative update rules to optimise the loss function. The modelling of the bias is tailored specifically to the problem of single side effect prediction. This model is the left component of DCSE-twin, namely, DCSE-left, see Chapter 4.2.

"If I have seen further it is by standing on the shoulders of Giants."

Isaac Newton

2 Related Work

In this chapter, I review several methods that have been proposed for the problem of predicting adverse DDIs. These methods will be used to compare the performance of my approach, DCSE-twin, against the state-of-the-art. The approaches in this chapter have been carefully selected based on two criteria, they have the best predictive performance; and they are highly representative of the different type of methods or technologies applied to this problem.

The availability of datasets that compile DDI information made possible the surge of computational methods that tackle this problem. There are two datasets that influenced the field the most, namely, TWOSIDES [23] and DrugBank [29]. TWOSIDES is a collection of significant drug-event associations from the AERS, which collects reports from doctors, patients and drug companies. DrugBank, on the other hand, contains information about DDIs that have some literature-based evidence. The types of DDIs are also different for both datasets. TWOSIDES includes the specific adverse side effect that occurs due to the co-administration of the two drugs, for example, the side effect "blindness" may be caused by the interaction between *vardenafil* and *tadalafil*. In DrugBank, the DDIs have different types and levels of specificity. They are categorised in pharmacokinetic and pharmacodynamic drug interactions. The nature and biological implications of these two datasets are different, and a distinction should be made when predicting for one or the other dataset.

Initially, various machine learning methods were developed to identify if two drugs will interact or not. Zhang et al. [30] propose a label propagation framework to predict DDIs based on clinical side effects. Kastrin et al. [31] frame the problem as a binary classification task on networks of potential DDIs and use link prediction techniques for predicting unknown interactions between drugs. Many other methods have been proposed, and most of them are based on drug similarities, such as interaction profiles, side effects and target similarities, 2D and 3D molecular structure, etc.[32, 33, 30, 34, 35, 36].

General DDI prediction approaches can be useful to derive broad rules for describing drug interactions at the cellular level, however, they can't be used directly to guide drug combination therapies [1]. These methods characterise DDIs through scores that can be interpreted as the probability of an interaction between the two drugs. However, these methods lack the ability to answer a more interesting question: *what are the specific type of adverse effect that arises from the interaction?* The methods presented in this chapter are focused on this important question.

There are several different adverse DDI prediction methods that have been proposed in the past years. Nevertheless, I have encountered a number of limitations with the existing methods. One of them is a lack of a systematic comparison of state-of-the-art methods under the same experimental setting. Another important piece missing is the lack of a biologically inspired research question that drives the experiments. Most of the methods perform a training/testing split that better suits their method, and overlook the usefulness of their setup. In Chapter 5, I present a comparison of my approach to several state-of-the-art methods under different experimental settings and provide a meaningful research question that helps elucidate the purpose of each experiment.

2.1 Modelling polypharmacy side effects with graph convolutional networks - Decagon

The work by Zitnik et al. [1] presents Decagon, an approach for modelling side effects caused by combinations of two drugs. This approach constructs a multimodal graph of protein-protein interactions, drug-protein target interactions, and the polypharmacy side effects, represented as drug-drug interactions, where each side effect is an edge of a different type.

Each drug-drug interaction is labelled by a different edge type, which signifies the type of the side effect. The authors develop a multirelational edge prediction

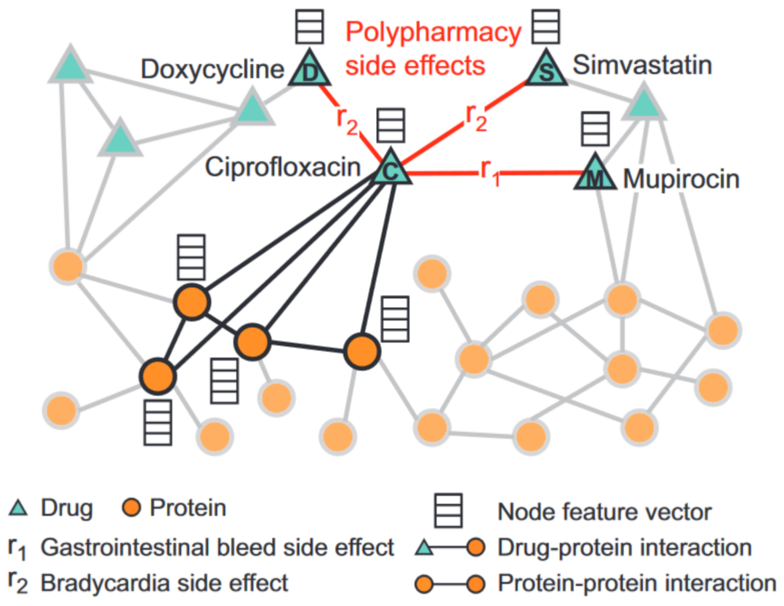

Polypharmacy side effects
Doxycycline
D
Simvastatin
S
r_2
r_2
Ciprofloxacin
C
r_1
Mupirocin
M
Drug
Protein
Node feature vector
r_1 Gastrointestinal bleed side effect
Drug-protein interaction
r_2 Bradycardia side effect
Protein-protein interaction

FIGURE 2.1: An example graph of polypharmacy side effects derived from genomic and patient population data (Figure is taken from [1]). A multimodal graph consists of protein–protein interactions, drug–protein targets and drug–drug interactions encoded by 964 different polypharmacy side effects (i.e. edge types $r_i, i = 1; ...; 964$). Side information is integrated into the model in the form of additional protein and drug feature vectors. Highlighted network neighbours of Ciprofloxacin (node C) indicate this drug targets four proteins and interacts with three other drugs. The graph encodes information that Ciprofloxacin (node C) taken together with Doxycycline (node D) or with Simvastatin (node S) increases the risk of bradycardia side effect (side effect type r_2), and its combination with Mupirocin (M) increases the risk of gastrointestinal bleed side effect r_1. The graph representation are used to develop Decagon, a graph convolutional neural model of polypharmacy side effects. Decagon predicts associations between pairs of drugs and side effects (shown in red) with the goal of identifying side effects, which cannot be attributed to either individual drug in the pair.

model that uses the multimodal graph to predict DDIs as well as their types. The polypharmacy side effect modelling is cast as a multirelational link prediction problem on a multimodal graph encoding drug, protein and side effect relationships (see Fig 2.1). These relationships are represented by a graph $G = (\mathcal{V}, \mathcal{R})$ with N nodes (e.g. proteins, drugs) $v_i \in \mathcal{V}$ and labelled edges (relations) (v_i, r, v_j), where r is the edge type (relation type): (i) physical binding between two proteins, (ii) a target relationship between a drug and a protein or (iii) a particular type of a side effect between two drugs. They consider 964 different relation types between drugs (i.e. side effects). *Note: after inspecting the dataset they made available, the actual number is 963.*

In addition, they allow for the inclusion of side information in the form of additional node features. Different nodes (drugs, proteins) can have different number of node features, given by real-valued feature vectors $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N$ assigned to every node in the graph.

The polypharmacy side effect prediction task considers the problem of identifying associations between drug pairs and side effects. These associations are limited to only those that cannot be attributed to either drug alone. Using the graph G, the task is to predict edge types between drug nodes. Given a drug pair (v_i, v_j), the aim is to determine how likely an edge $e_{ij} = (v_i, r, v_j)$ of type r belongs to $\mathcal{R}$, meaning that concurrent use of drugs v_i and v_j [i.e. the use of a drug combination (v_i, v_j)] is associated with a polypharmacy side effect of type r in the human patient population.

To this aim, the authors develop a non-linear, multi-layer convolutional graph neural network model Decagon that operates directly on graph G. Decagon has two main components:

- an encoder: a graph convolutional network operating on G and producing embeddings for nodes in G (Fig. 2.2A), and
- a decoder: a tensor factorisation model using these embeddings to model polypharmacy side effects (Fig. 2.2B).

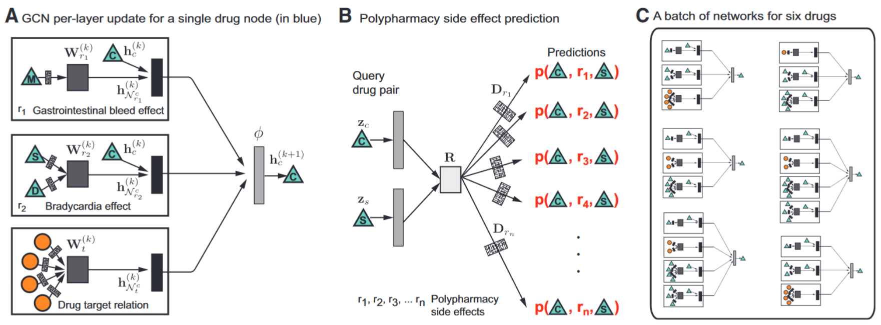

A GCN per-layer update for a single drug node (in blue)
B Polypharmacy side effect prediction
C A batch of networks for six drugs
r1 Gastrointestinal bleed effect
r2 Bradycardia effect
Drug target relation
Query drug pair
Predictions
Polypharmacy side effects

FIGURE 2.2: Overview of Decagon model architecture (Figure taken from [1]). (A) Decagon encoder. Shown is a per-layer update for a single graph node (a drug node representing Ciprofloxacin based on the small example input graph in Fig. 2.1). Hidden state activations from neighboring nodes $\mathcal{N}_r^c$ are gathered and then transformed for each relation type individually and passed through a non-linear activation function (i.e. ReLU) to produce a hidden state of node v_c in the $(k+1)$-th layer, $\mathrm{h}_c^{(k+1)}$. This per-node update is computed in parallel with shared parameters across the whole graph. (B) For every relation, the Decagon decoder takes pairs of embeddings (e.g. hidden node representations z_c and z_s representing Ciprofloxacin and Simvastatin) and produces a score for every (potential) edge in the graph. Shown is the decoder for polypharmacy side effects relation types. (C) A batch of neural networks that compute embeddings of six drug nodes in the input graph. In Decagon, neural networks differ from node to node but they all share the same set of relation-specific trainable parameters [i.e. the parameters of the encoder and decoder; see Equations 2.1 and 2.2]. That is, rectangles with the same shading patterns share parameters, and thin rectangles with black and white shading patterns denote densely connected neural layers.

2.1.1 *Decagon* encoder

Let's first describe the graph encoder model. It takes as input a graph G and additional node feature vectors $\mathbf{x}_i$, and produces a node d-dimensional embedding $\mathrm{z}_i \in \mathbb{R}^d$ for every node (drug or protein) in the graph. The idea is that *Decagon* learns how to transform and propagate information, captured by node feature vectors, across the graph. Convolutional operators are learned that propagate and transform information across different parts of the graph and across different relation types. For a given node *Decagon* performs transformation/aggregation operations on feature vectors of its neighbours. This way *Decagon* only takes into account the first-order neighbourhood of a node and applies the same transformation across all locations in the graph. Successive application of these operations then effectively convolves information across the K-th order neighbourhood (i.e. embedding of a node depends on all the nodes that are at most K steps away), where K is the number of successive operations of convolutional layers in the neural network model. In each layer, Decagon propagates latent node feature information across edges of the graph, while taking into account the type (relation) of an edge. A single layer of this neural network model takes the following form:

$$\mathbf{h}_i^{(k+1)} = \phi\left(\sum_r \sum_{j \in \mathcal{N}_r^i} c_r^{ij} \mathbf{W}_r^{(k)} \mathbf{h}_j^{(k)} + c_r^i \mathbf{h}_i^{(k)}\right) \tag{2.1}$$

where $\mathbf{h}_i^{(k)} \in \mathbb{R}^{d(k)}$ is the hidden state of node v_i in the k-th layer of the neural network with $d^{(k)}$ being the dimensionality of this layer's representation, r is a relation type and matrix $\mathbf{W}_r^{(k)}$ is a relation-type specific parameter matrix. Here, ϕ denotes a non-linear element-wise activation function (i.e. a rectified linear unit), which transforms the representations to be used in the layer of the neural model, c_r^{ij} and c_r^i are normalisation constants, which are chosen to be symmetric $c_r^{ij} = 1/\sqrt{|\mathcal{N}_r^i||\mathcal{N}_r^i|}$ and $c_r^i = 1/|\mathcal{N}_r^i|$ with $\mathcal{N}_r^i$ denoting the set of neighbours of node v_i under relation r. Importantly note that the sum in Equation 2.1 ranges only over the neighbours $\mathcal{N}_r^i$ of a given node i and thus the computational architecture (i.e. the neural network) is different for every node. Figure 2.2A shows an example of a per-layer convolutional update Equation 2.1 for node C from Figure 2.1. And, Figure 2.2C then illustrates that different nodes have different structures of neural networks, because each node's network neighbourhood is different.

The overall encoder then takes the following form. The authors stack K layers as defined in Equation 2.1 such that the output of the previous layer becomes the input to the next layer. The input to the first layer are node feature vectors, $\mathbf{h}_i^{(0)} = \mathbf{x}_i$, or unique one-hot vectors for every node in the graph if no features are present.

2.1.2 *Decagon* decoder

The encoder maps each node $v_i \in \mathcal{V}$ to an embedding, a real-valued vector representation $\mathbf{z}_i \in \mathbb{R}^d$, where d is the dimensionality of node representations.

The goal of the decoder is to reconstruct labelled edges in G by relying on learned node embeddings and by treating each label (edge type) differently. In particular, decoder scores a (v_i, r, v_j)-triple through a function g whose goal is to assign a score $g(v_i, r, v_j)$ representing how likely it is that drugs v_i and v_j are interacting through a relation/side effect type r (Fig. 2.2C). Using embeddings for nodes i and j returned by *Decagon's* encoder z_i and z_j, the decoder predicts a candidate edge (v_i, r, v_j) through a factorised operation:

$$g(v_i, r, v_j) = \begin{cases} \mathbf{z}_i^T \mathbf{D}_r \mathbf{R} \mathbf{D}_r \mathbf{z}_j & \text{if } v_i \text{ and } v_j \text{ are drugs} \\ \mathbf{z}_i^T \mathbf{M}_r \mathbf{z}_j & \text{if } v_i \text{ and } v_j \text{ are both proteins, or,} \\ & v_i \text{ and } v_j \text{ are a protein and a drug} \end{cases} \tag{2.2}$$

followed by the application of a sigmoid function σ to compute probability of edge (v_i, r, v_j) :

$$p_r^{ij} = p\left((v_i, r, v_j) \in \mathcal{R}\right) = \sigma\left(g\left(v_i, r, v_i\right)\right) \tag{2.3}$$

Decagon's decoder distinguishes between two different cases:

1. When v_i and v_j are drug nodes, the decoder g in Equation 2.2 assumes a global model of drug-drug interactions (i.e. $\mathbf{R}$) whose variation and importance across polypharmacy side effects are described by side-effect-specific diagonal factors (i.e. $\mathbf{D}_r$). Here, $\mathbf{R}$ is a trainable parameter matrix of shape $d \times d$ that models global DDIs across all possible polypharmacy side effects. Additionally, in Decagon, every relation r representing a different polypharmacy side effect is associated with a diagonal $d \times d$ matrix D_r modelling the importance of each dimension in z_i towards side effect r.

2. When v_i and v_j are not both drug nodes, the decoder g in Equation 2.2 employs a bilinear form to decode edges from node embeddings. More precisely, in that case, the decoding function g is associated with a trainable parameter matrix $\mathbf{M}_r$ of shape $d \times d$ that models interactions between every two dimensions in $\mathbf{z}_i$ and $\mathbf{z}_j$. The predicted edge probability is then computed using a bilinear form (Equation 2.2) followed by the application of a sigmoid function (Equation 2.3).

2.1.3 *Decagon* model training

During the model training, *Decagon's* model parameters are optimised using the cross-entropy loss:

$$J_r(i,j) = -\log p_r^{ij} - \mathbb{E}_{n \sim P_r(j)} \log \left(1 - p_r^{in}\right), \tag{2.4}$$

to encourage the model to assign higher probabilities to observed edges (v_i, r, v_j) than to random non-edges, where $\mathbb{E}_{n \sim P_r(j)}$ is the expectation when n follows the negative sampling distribution P_r. As in previous studies [26, 37], the model is estimated through negative sampling. For each drug-drug edge (v_i, r, v_j) (i.e. a positive example) in the graph, a random edge (v_i, r, v_n) (i.e. a negative example) is sampled by randomly choosing node v_n. This is achieved by replacing node v_j in edge (v_i, r, v_j) with node v_n that is selected randomly according to a sampling distribution P_r [26]. Considering all edges, the final loss function in Decagon is:

$$J = \sum_{(v_i, r, v_j) \in \mathcal{R}} J_r(i, j). \tag{2.5}$$

Decagon is trained for a maximum of 100 epochs using the Adam optimiser with a learning rate of 0.001 and early stopping with a window size of 2.

2.1.4 *Decagon* experimental setup

Decagon is trained on the TWOSIDES dataset, focusing on predicting the 963 commonly occurring types of polypharmacy side effects that each occurred in at least 500 drug combinations.

For each polypharmacy side effect type, drug pairs associated with that side effect are split into training, validation and test sets. Ensuring that the validation and test sets each include 10% of drug pairs. For each side effect type, 80% of drug pairs are used to train the model and 10% to select model parameters. The task is then to predict pairs of drugs that are associated with each side effect type. *Decagon* assigns a probability for every drug pair and for every side effect type, that a given pair is associated with the given side effect.

2.2 DeepWalk

DeepWalk [2] is an approach for learning latent representations of vertices in a network. These representations encode relations in a continuous vector space, which can be exploited by statistical models. DeepWalk uses local information obtained from truncated random walks to learn latent representations by treating walks as the equivalent of sentences. DeepWalk was not developed explicitly for the prediction of polypharmacy side effects. However, it was applied by Zitnik et al. in the *Decagon* paper, as a competing method. I choose this method because, similarly to my approach, it learns representations for drugs, but follows a graph-based approach.

2.2.1 Problem formulation

The authors consider the problem of classifying members of a social network into one or more categories. Let $G = (V, E)$, where V represent the members of the network, E are their connections, $E \subseteq (V \times V)$, and $G_L = (V, E, X, Y)$ is a partially labelled social network, with attributes $X \in \mathbb{R}^{|V| \times S}$ where S is the size of the feature space for each attribute vector, and $Y \in \mathbb{R}^{|V| \times |\mathcal{Y}|}, \mathcal{Y}$ is the set of labels.

In a traditional machine learning classification setting, the aim is to learn a hypothesis H that maps elements of X to the labels set $\mathcal{Y}$. In this case, the significant information about the dependence of the examples embedded in the structure of G can be used to achieve superior performance.

The goal is to learn $X_E \in \mathbb{R}^{|V| \times d}$, where d is small number of latent dimensions. These low-dimensional representations are distributed; meaning each social phenomenon is expressed by a subset of the dimensions and each dimension contributes to a subset of the social concepts expressed by the space. This is the same concept

I apply in my approach. Using these structural features, the attributes space will be augmented to help the classification decision. These features are general and can be used with any classification algorithm (including iterative methods). However, the greatest utility of these features is their easy integration with simple machine learning algorithms. They scale appropriately in real-world networks. This is why DeepWalk can be applied to the problem of polypharmacy side effect prediction. DeepWalk takes a graph as input and produces a latent representation as output. Once the distributed representations are learned by the method, any state-of-the-art machine learning algorithm, such as Support Vector Machines (SVMs) or Random Forests (RFs), can be applied using the representations as features.

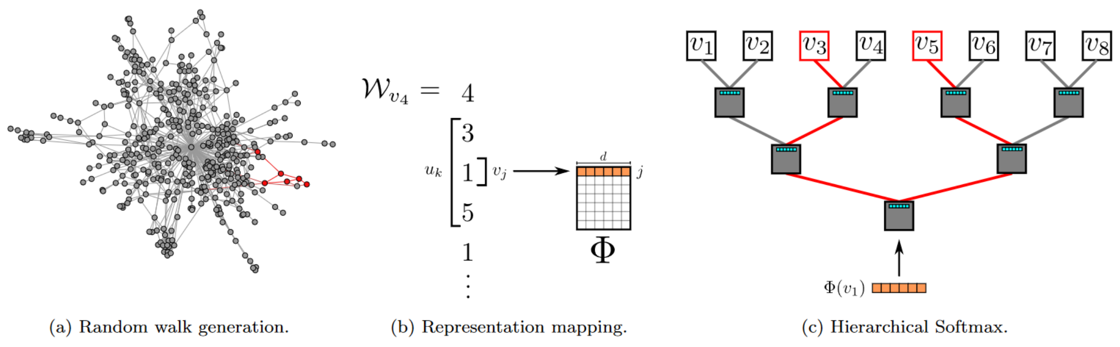


FIGURE 2.3: Overview of DeepWalk (Figure taken from [2]). A window of length $2w+1$ is slid over the random walk $\mathcal{W}_{v_4}$, mapping the central vertex v_1 to its representation $\Phi(v_1)$. Hierarchical Softmax factors out $\Pr(v_3 \mid \Phi(v_1))$ and $\Pr(v_5 \mid \Phi(v_1))$ over sequences of probability distributions corresponding to the paths starting at the root and ending at v_3 and v_5. The representation Φ is updated to maximise the probability of v_1 co-occurring with its context $\{v_3, v_5\}$.

2.2.2 The DeepWalk method

The only required input for any language modelling algorithm is a corpus and a vocabulary $\mathcal{V}$. DeepWalk takes a set of short truncated random walks as its own corpus, and the graph vertices as its own vocabulary ($\mathcal{V} = V$).

The DeepWalk algorithm consists of two main components; first a random walk generator, and second, an update procedure. The random walk generator takes a graph G and samples uniformly a random vertex v_i as the root of the random walk $\mathcal{W}_{v_i}$. A walk samples uniformly from the neighbours of the last vertex visited until the maximum length (t) is reached (Fig. 2.3a). In practice, the implementation specifies the number of random walks γ of length t to start at each vertex.

ALGORITHM 1: DeepWalk(G, w, d, γ, t)

```
Input  : graph $G(V, E)$
         window size $w$
         embedding size $d$
         walks per vertex $\gamma$
         walk length $t$
Output: matrix of vertex representations $\Phi \in \mathbb{R}^{|V| \times d}$
1 Initialisation: Sample $\Phi$ from $\mathcal{U}^{|V| \times d}$
2 Build a binary Tree $T$ from $V$
3 for $i = 0$ to $\gamma$ do
4     $\mathcal{O} = \text{Shuffle}(V)$
5     foreach $v_i \in \mathcal{O}$ do
6         $\mathcal{W}_{v_i} = \text{RandomWalk}(G, v_i, \text{t})$
7         $\text{SkipGram}(\Phi, \mathcal{W}_{v_i}, w)$
8     end
9 end
```

Lines 3-9 in Algorithm 1 show the core of this approach. The outer loop specifies the number of times, γ, that should start random walks at each vertex. Each iteration can be thought of as making a *pass* over the data and sampling one walk per node during this pass. At the start of each pass, a random ordering is generated to traverse the vertices. The SkipGram algorithm [26] is used to update these representations in accordance with the following objective function:

$$\underset{\Phi}{\text{minimize}} \quad -\log \Pr\left(\{v_{i-w}, \cdots, v_{i+w}\} \setminus v_i \mid \Phi(v_i)\right) \tag{2.6}$$

ALGORITHM 2: SkipGram($\Phi, \mathcal{W}_{v_i}, w$)

```
1 foreach $v_j \in \mathcal{W}_{v_i}$ do
2     foreach $u_k \in \mathcal{W}_{v_i}[j - w : j + w]$ do
3         $J(\Phi) = -\log \Pr(u_k \mid \Phi(v_j))$
4         $\Phi = \Phi - \alpha * \frac{\partial J}{\partial \Phi}$
5     end
6 end
```

SkipGram

SkipGram is a language model that maximises the co-occurrence probability among the words that appear within a window w in a sentence. It approximates the conditional probability in Eq. 2.6 using an independence assumption as the following:

$$\Pr\left(\{v_{i-w}, \cdots, v_{i+w}\} \setminus v_i \mid \Phi(v_i)\right) = \prod_{\substack{j=i-w \\ j \neq i}}^{i+w} \Pr\left(v_j \mid \Phi(v_i)\right) \tag{2.7}$$

Algorithm 2 iterates over all possible collocations in a random walk that appear within the window w (lines 1-2). For each, a vertex v_j is mapped to its current representation vector $\Phi(v_j) \in \mathbb{R}^d$ (see Fig. 2.3b). Hierarchical Softmax is used to approximate the probability distribution.

The model parameter set is $\theta = \{\Phi, \Psi\}$ where the size of each is $O(d|V|)$. Stochastic gradient descent (SGD) is used to optimise these parameters (Line 4, Algorithm 2). The derivatives are estimated using the back-propagation algorithm. The learning rate α for SGD is initially set to 2.5% at the beginning of the training and then decreases linearly with the number of vertices that are seen so far.

For the problem of polypharmacy side effect prediction, once the distributed representations are learned for each drug, a logistic regression classifier is trained for each side effect. This is the same methodology as described in [1], and the same experimental setup can be applied.

2.3 Predicting Polypharmacy Side-effects Using Knowledge Graph Embeddings - TriVec

TriVec is a knowledge graph-based approach applied to the problem of polypharmacy side effect prediction [3]. This method uses multi-part embedding vectors to predict the side effects caused by the interaction between two drugs. This approach models drug pairs and their side effects as a knowledge graph, where interacting drugs are modelled as nodes and their corresponding side effects are modelled as edges. A tensor factorisation-based knowledge graph embedding model is used to learn vector representations for both drugs and side effects. These learned embeddings are used to predict unknown DDIs. This approach has been shown to achieve superior performance when compared to Decagon [3]. It also learns representations for both drugs and side effects, as my own approach. For these reasons, I choose TriVec as the main competing method for my work.

2.3.1 TriVec method

The knowledge graph embedding model that is used, follows a generative approach to learning low-rank embedding vectors for the entities and relations. The TriVec model represents each entity and relation using three embedding vectors such that the embedding of entity i is $\Theta_E(i) = \{e_i^1, e_i^2, e_i^3\}$ where all embedding vectors have the same size K (a user-defined embedding size). Similarly, the embedding of relation j is $\Theta_R(j) = \{w_j^1, w_j^2, w_j^3\}$. Where e^m and w^m denote the m embedding vector for the entity or the relation, and $m \in \{1, 2, 3\}$. The embeddings in the TriVec model are initially with random values generated by the Glorot uniform random generator. The embedding vectors are then updated during the training procedure to provide optimised scores for the knowledge graph facts.

TriVec first initialises its embedding with random noise. Then, it updates them by iterative learning on the training data. The learning pipeline of the model learns the embeddings of entities and relations by minimising a negative softmax log-loss that maximises the scores of true facts and minimises the scores of unknown facts. This loss is defined as follows:

$$\begin{aligned}\mathcal{L}_{\text{spo}}^{\text{TriModel}} &= -\phi_{spo'} + \log\left(\sum_{o'} \exp\left(\phi_{spo'}\right)\right) - \phi_{s'po} + \log\left(\sum_{s'} \exp\left(\phi_{s'po}\right)\right) \\ &+\frac{\lambda}{3}\sum_{k=1}^{K}\sum_{m=1}^{3}\left(|e_s^m|^3 + \left|w_p^m\right|^3 + |e_o^m|^3\right),\end{aligned} \tag{2.8}$$

where K denotes the embedding size, $\phi_{spo'}$ denotes the score of the (s, p, o') triple, x' represents an entity $e : e \neq x, e \in \mathcal{E}$, e_i^m is the embedding part m of the entity embedding $\Theta_E(i)$, w_i^m is the embedding part m of the relation embedding $\Theta_R(i)$, $\phi_{s'po}$ denotes the score of the triple $(s', p, o), m$ denotes the embedding part index, λ denotes a configurable regularisation weight parameter and $|x|$ is the absolute of x. The term $\frac{\lambda}{3}\sum_{k=1}^{K}\sum_{m=1}^{3}\left(|e_s^m|^3 + \left|w_p^m\right|^3 + |e_o^m|^3\right)$ is the nuclear 3-norm, which is a regularisation term that enhances model generalisation over datasets with large entity vocabularies.

The scores of the TriVec model are computed using an embeddings interaction function (scoring function) that is defined as follows:

$$f_{\text{TriModel}}\left(s, p, o, \Theta\right) = \sum^{K} e_s^1 w_p^1 e_o^3 + e_s^2 w_p^2 e_o^2 + e_s^3 w_p^3 e_o^1. \tag{2.9}$$

It uses a set of three interactions: one symmetric interaction: $\left(e_s^2 w_p^2 e_o^2\right)$ and two asymmetric interactions: $(e_s^1 w_p^1 e_o^3)$ and $(e_s^3 w_p^3 e_o^1)$.

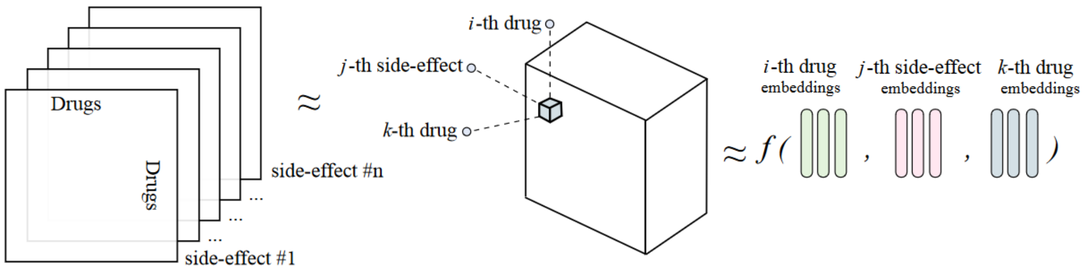


FIGURE 2.4: Flow diagram of the scoring function of TriModel applied to polypharmacy side-effects drug combinations (Figure taken from [3]). The subject (s), and the object (o) represent the drug combination while their corresponding polypharmacy side-effect is represented as the relation (r). All these components are represented using three embedding vectors of size k. The score of a triple (s, p, o) is defined as $f(s, p, o) = \sum_k c_k$, where $c = i_1 + i_2 + i_3, i_1 = s1 \cdot r_1 \cdot o3, i2 = s_2 \cdot r_2 \cdot o_2$, and $i_3 = s_3 \cdot r_3 \cdot o_1$.

The TriVec model is a tensor factorisation model that solves the problem of link prediction as 3D tensor completion. The tensor dimensions represent entities and relations. In the task of predicting polypharmacy side-effects, the drug combinations are modelled as the subjects and objects of triples while the corresponding polypharmacy side-effects are modelled as relations. In the training process, the model learns from the different types of assertions such as protein-protein interactions, drug-protein interactions, drug-drug interactions, single drug side effects and polypharmacy side effects. This allows the model to learn efficient embeddings for the components corresponding to the different entities and relations in the knowledge graph. In the prediction phase, the TriVec model then learns the probability of polypharmacy side-effects associations by completing a 3D tensor of drugs and polypharmacy side-effects as shown in Figure 2.4.

The evaluation protocol is the same as in the work by Zitnik et al., as they compare this method against Decagon. A detailed explanation for this procedure can be found in Chapter 5 of this thesis. The results obtained greatly surpass the performance of Decagon and show how the learned embeddings can be used to enhance the predictive performance for graph-based methods.

2.4 Deep learning improves the prediction of drug–drug and drug–food interactions

The last method I present is DeepDDI, by Ryu et al. [4]. This is a computational framework that uses names of drug-drug or drug-food constituent pairs and their structural information as inputs to accurately generate DDI types as outputs of human-readable sentences. DeepDDI uses Deep Neural Networks (DNN) to predict DDI types using the DrugBank gold standard DDI dataset. This is a method that was developed for a different DDI dataset. However, this approach uses deep neural networks at the core of its predictions, making it an excellent baseline to compare against my work.

The input for the DNN starts with the structural information provided in the Simplified Molecular-Input Line-Entry System (SMILES)[38] format. This representation was used to generate a feature vector called Structural Similarity Profile (SSP). SSP is devised to effectively capture unique structural features of a given drug and to associate this feature with a set of the reported DDI types. SSP contains pairwise structural similarity scores obtained from the comparison between the input drug and all the 2,159 approved drugs, in the DrugBank dataset, as a fixed comparison target. The SSP is generated for each drug in the input drug pair. The similarity is measured by the Tanimoto coefficient, which is defined as the number of common chemical fingerprints divided by the number of all the chemical fingerprints of the two drugs being compared (Fig. 2.5). The chemical fingerprints of each drug are calculated by using the Extended-Connectivity Fingerprint of diameter 4 (ECFP4). The dimension of the SSPs is reduced using Principal Component Analysis (PCA) and keeping 50 principal components (90% of variance retained). The two SSPs of each input drug are combined and used as the input for the DNN of DeepDDI, as observed in Figure 2.6.

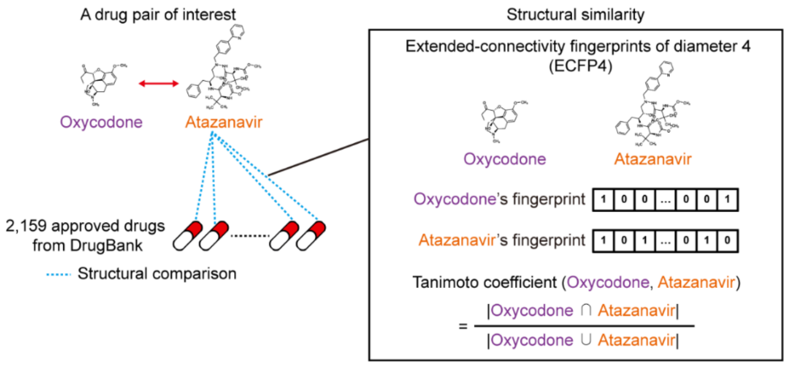

A drug pair of interest
Oxycodone
Atazanavir
2,159 approved drugs from DrugBank
Structural comparison
Structural similarity
Extended-connectivity fingerprints of diameter 4 (ECFP4)
Oxycodone
Atazanavir
Oxycodone's fingerprint 1 0 0 ... 0 0 1
Atazanavir's fingerprint 1 0 1 ... 0 1 0
Tanimoto coefficient (Oxycodone, Atazanavir)
= |Oxycodone ∩ Atazanavir| / |Oxycodone ∪ Atazanavir|

Figure 2.5: Concept of Structural Similarity Profile (SSP) (Figure taken from [4]). SSP is a feature vector for an input drug in the pair and contains pairwise structural similarity scores obtained from the comparison between the input drug and the 2,159 approved drugs reported at DrugBank. The structural similarity was calculated using the Tanimoto coefficient with extended-connectivity fingerprints of diameter 4 (ECFP4).

The DNN of DeepDDI is designed to be a multi-label classification model that predicts multiple DDI types for a given drug pair. This is modelled this way since drug pairs can have multiple DDI types. DeepDDI is built to predict 86 different DDI type sentences, this means that it has the same number of output neurons. These output neurons' activity values range between 0-1 and can be interpreted as the probability of the pair interacting to cause that DDI type. The DDN is trained by minimising the cross-entropy as the loss function. An overview of DeepDDI can

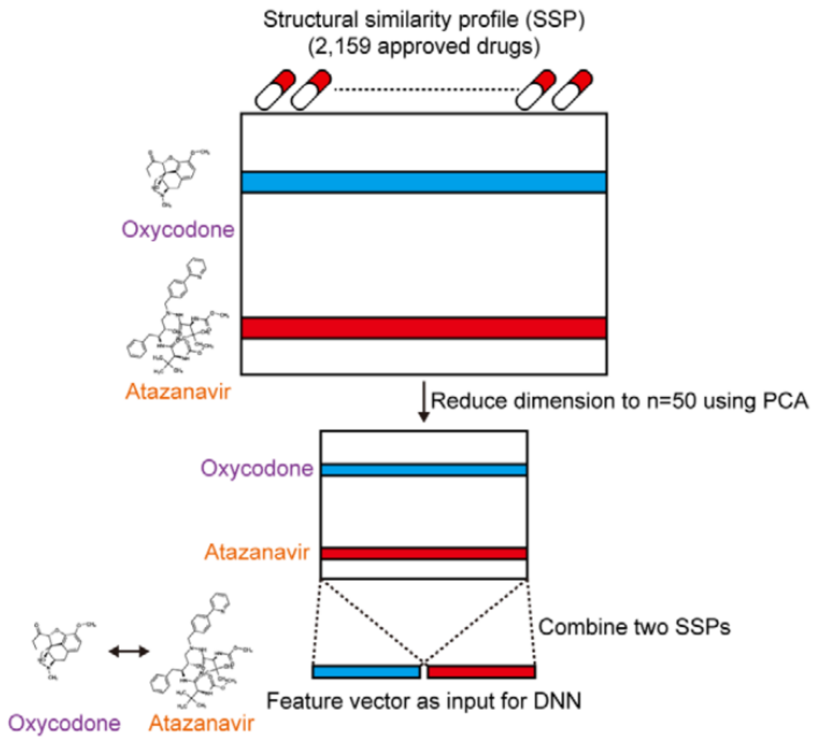

Structural similarity profile (SSP)
(2,159 approved drugs)
Oxycodone
Atazanavir
Reduce dimension to n=50 using PCA
Oxycodone
Atazanavir
Combine two SSPs
Feature vector as input for DNN
Oxycodone
Atazanavir

FIGURE 2.6: Process of using the SSPs as input for the DNN (Figure taken from [4]). Upon generation of the SSP for each drug in pair (Fig 2.5), the dimension of each SSP is reduced to 50 using PCA. The two SSPs with the reduced dimension are subsequently combined into a single SSP to represent each drug pair as a feature vector. The combined SSP is used as an input for the DNN of DeepDDI.

be observed in Figure 2.7.

For the evaluation setting, the drug pairs associated with the DDI types in the gold-standard DrugBank DDI dataset are split into training (60%), validation (20%) and testing (20%). A total of 192,284 DDIs form the dataset used in this study.

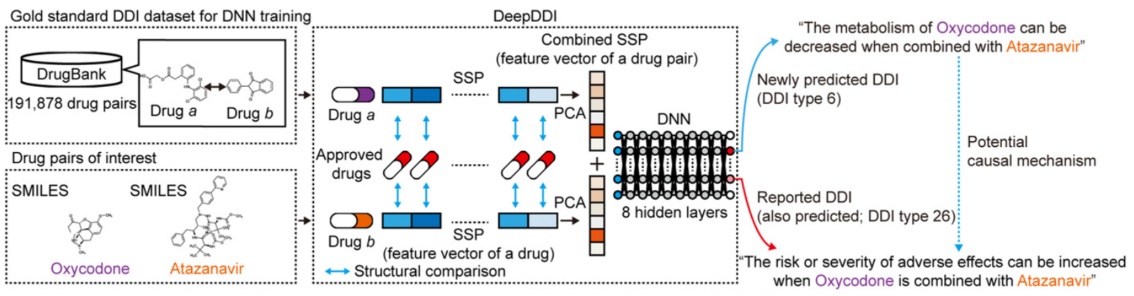

Gold standard DDI dataset for DNN training
DrugBank
191,878 drug pairs
Drug a
Drug b
Drug pairs of interest
SMILES
SMILES
Oxycodone
Atazanavir
DeepDDI
Combined SSP
(feature vector of a drug pair)
SSP
Drug a
PCA
DNN
Approved drugs
8 hidden layers
PCA
Drug b
SSP
(feature vector of a drug)
Structural comparison
"The metabolism of Oxycodone can be decreased when combined with Atazanavir"
Newly predicted DDI
(DDI type 6)
Potential causal mechanism
Reported DDI
(also predicted; DDI type 26)
"The risk or severity of adverse effects can be increased when Oxycodone is combined with Atazanavir"

Figure 2.7: Overall scheme of the DeepDDI approach (Figure taken from [4]). DeepDDI consists of the structural similarity profile (SSP) generation pipeline and deep neural network (DNN). DeepDDI accepts chemical structures (in SMILES describing the structure of a chemical compound) and names of drugs in pairs as inputs and predicts their potential drug-drug interaction (DDI) types as outputs in human-readable sentences having the input drug names. DNN of DeepDDI is a multilabel classification model that can predict multiple DDI types at the same time for a given drug pair. To develop DeepDDI, a gold standard DDI dataset covering 191,878 drug pairs was obtained from DrugBank, and used to train the DNN of DeepDDI. A single, combined SSP (feature vector of a drug pair) is generated for each input drug pair. DeepDDI has many implications such as the prediction of potential causal mechanisms for the adverse drug events (ADEs) of a drug pair of interest (blue dotted arrow) using the output sentences.

2.5 Final remarks

In this chapter, I presented an overview of the state-of-the-art in DDI prediction. The first methods developed to this end focused on predicting whether two drugs interact or not. These approaches frame DDI prediction as a binary classification problem. However, there is a detailed type of polypharmacy side effect (e.g., hepatic failure, cough, high blood pressure, lower metabolism, etc.) associated with each DDI. There are several methods that were developed for this problem [3, 39, 40, 41, 42]. I selected only the most relevant ones as competing methods in my work, since these approaches are highly representative of their own categories of methods and serve as good baselines to compare against my work. Also, they were chosen based on their relevance to the field and impact of the publication venue.

A limitation of the current literature is that there is a lack of a unified and standard evaluation protocol. Different methods adopt different evaluation settings that best suit their own architectures and modelling of the problem. Another issue is that the biological reasoning behind the experiments is often overlooked or missing altogether. A key limitation, that was identified by Qian et al., is the fact that most current methods can't generalise their performance on drug pairs where neither drug is seen during training [43]. Qian et al., are the firsts to introduce a more realistic experimental setting that addresses the *cold start* problem of predicting drugs completely unseen during training. In this work, the authors leverage genetic interaction features and introduce a novel training scheme for DDI prediction in the DrugBank dataset. One of the contributions of their work is to showcase the strength of the genetic interaction profiles as features for the prediction of DDIs and the novel training scheme. They make use of XgBoost as the machine learning method to learn a model for the predictions. I don't compare against the work by Qian et al. because they focus on the problem of general DDI prediction. Also, the number of drugs is reduced by the constraint of having the features available. Nevertheless, I include a comparison against the XgBoost method (see Chapter 5.1) but using other type of features that are available for all the drugs in the TWOSIDES dataset.

In methods such as Decagon, the drug pairs are split by polypharmacy side effects. This is what makes the most sense for graph-based methods, since the side effect is the edge type. The testing can then be interpreted as sampling edges from different graphs. However, it is somewhat far from a real-world scenario. More often than not, DDIs are researched for a new drug or a new drug combination during the clinical trials in the drug development pipeline. And these graph-based methods depend on known polypharmacy side effects to make efficient predictions. They require rich information about the investigated drugs, i.e., known associations about interactions with other drugs. This means that these methods cannot make predictions for drugs with limited or no known assertions [3]. Moreover, based on the graph-nature of the method, it would be near impossible to modify it to account for unseen drugs or side effects.

My work addresses the limitations found in the current literature on polypharmacy side effect prediction. I introduce well-defined experimental settings, found in Chapter 5.1. I also provide an intuition of the biological motivation behind each research question that the evaluation intends to address. Besides covering the most conventional evaluation protocol found in the literature of DDI prediction, the other experimental settings I propose are novel and provide a more realistic scenario that experimentalists can relate to. Lastly, I provide a sound comparison of my approach against the state-of-the-art methods presented in this chapter for each experimental setting.

"Progress is made by trial and failure; the failures are generally a hundred times more numerous than the successes, yet they are usually left unchronicled."

William Ramsay

3 Early Models

The goal of my work is to develop machine learning models to predict polypharmacy side effects, i.e. adverse effects caused by the co-administration of two drugs. In this chapter, I present the early models that were crucial to understanding the problem and realising some key biological aspects about the prediction of adverse DDIs. The objective of this chapter is to showcase what were the limitations and findings that led to the development of my final approach, DCSE-twin.

Galeano et al. [44] developed a linear novel matrix decomposition method that learns low-dimensional representations for drugs and side effects, and, for the first time, predicts the frequency of side effects. This prompted me to question if I could model the prediction of adverse DDIs, using a similar, linear, model. Specifically, *can I model the combination of two drugs with a linear model?*

To answer this question, first, I developed a matrix decomposition method that breaks down the DDI matrix (described in Chapter 3.2) into the product of three matrices, S, M, and H. The matrices S and H are fixed and contain the representations learned by Galeano et al. This method, attempts to linearly combine the individual representations of the two drugs and relate them to the representations of the side effects. The main finding from this model showed that a linear model is not sufficient to capture the combination of the representations of single drugs. To explore this notion further, I formulated a nonlinear approach to combine the pre-trained single drug representations. This approach uses a Mutli-Layer Perceptron (MLP) to combine the representations for the individual drugs and learn a representation for the drug pairs. The results from these early models, indicated that the drugs, in fact, are combining nonlinearly.

3.1 Materials

3.1.1 Drug-Drug Interaction Data

The adverse DDIs were taken from TWOSIDES [23], which contains information of 63,473 drug combinations and 1,318 side effect types with a total of 645 drugs. This data was mined from the FDA AERS (FAERS) for reports of adverse effects in patients receiving combinations of two drugs as medication. In Chapter 5.1.1 I discuss this dataset in more detail.

3.1.2 Drug and side effect representations

A recent work, developed in our lab by Galeano et al., [44] presents a novel matrix decomposition method that predicts the frequency of every side effect for every single drug in the SIDE Effect Resource (SIDER)[1] 4.1 [45] dataset. This decomposition results in a vector of latent variables representing each drug and a vector of latent variables representing each side effect. Galeano et al. show that these low-dimensional representations encode meaningful biological information of drug molecular and clinical activity. Furthermore, the individual components of the vectors are interpretable in terms of drug anatomical categories.

I intend to use the underlying biological information, encoded by the representations learned by Galeano et al., for the prediction of DDIs. The assumption here is that the relationship between the drug representations will indicate how the drugs interact with each other and which anatomical systems will be perturbed to cause a specific adverse effect.

The outcome of applying the method by Galeano et al. are two matrices of low-dimensional vectors. Let's define the matrix $W \in \mathbb{R}^{m \times k}$ that contains the drugs representations, and the matrix $H \in \mathbb{R}^{k \times n}$ formed by the side effects representations, where m is the number of drugs and n is the number of side effects. For the DDI prediction problem, the number of drugs is, $m = 432$, and the number of side effects is, $n = 544$. The total number of combinations is, $36,143$.

[1]This dataset contains information on marketed medicines and their recorded adverse drug reactions. The information is extracted from public documents and package inserts.

3.2 SMH Model

I first propose a linear model for the prediction of adverse DDIs. This is a matrix decomposition model that combines pre-trained single drug representations and learns a matrix that relates the representations of the two drugs with the pre-trained side effect representations. In other words, I propose a decomposition of the DDI matrix into the product of three matrices, S, M, and H. An overview of this approach can be observed in Figure 3.1.

First we represent the DDI information as a matrix $X \in \mathbb{R}^{p \times n}$ where p is the total number of drug combinations with at least one side effect associated to them. I use the pre-trained drugs latent vectors, contained in W, to represent each drug. Let $S \in \mathbb{R}^{p \times 2k}$ be a matrix where each row is a drug pair represented by the concatenation of the representations of the individual drugs, $s_i \in \mathbb{R}^{2k}$ where k is the length of each low-dimensional vector ($k = 10$). Accordingly, I fix the matrix $H \in \mathbb{R}^{k \times n}$ where each column represents the pre-trained representation of each side effect.

To relate these two matrices S and H the objective is to learn a new matrix $M \in \mathbb{R}^{2k \times k}$ that will reduce the concatenated representations, in S, back into the same latent space of length k. The matrix M will contain the learned representations for the drug combinations. I obtain the prediction for the polypharmacy side effects by applying the following linear model:

$$\hat{X} \approx SMH \tag{3.1}$$

where M is the solution of the following optimisation problem:

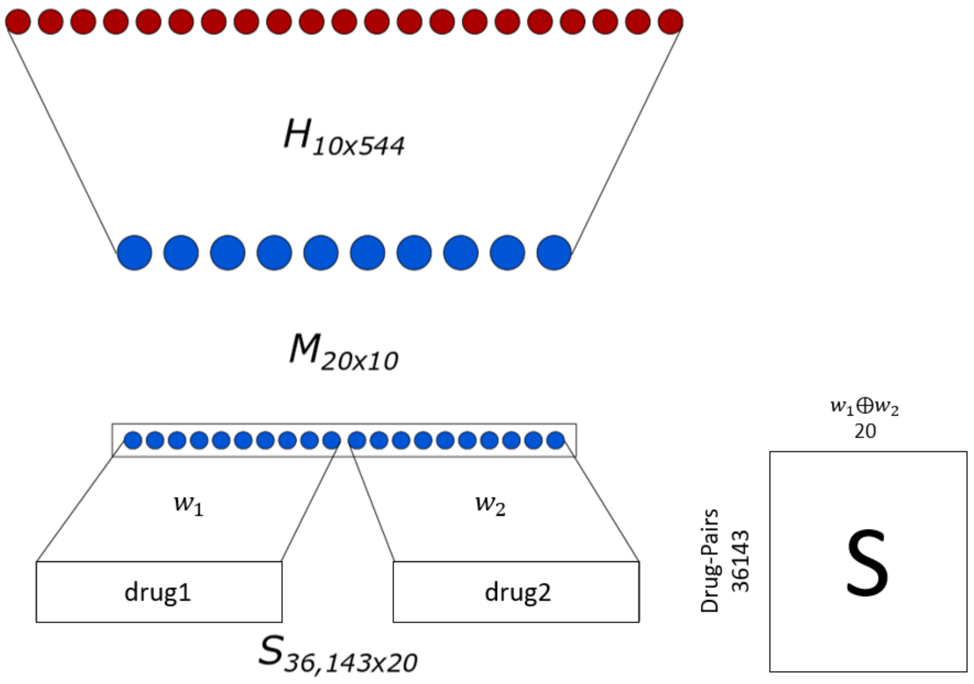


Figure 3.1: SMH model. This is a matrix decomposition linear model that learns a matrix M that relates the matrices S and H. S contains the pre-trained concatenated drug representations for every pair, and H is formed by the pre-trained representations for the side effects.

$$\min_{M} \mathcal{J}_{\text{SMH}}(M) = \frac{1}{2}\|X - SMH\|_F^2 + \frac{\beta}{2}\|M\|_F^2 \quad \text{subject to the non-negative constraint } M \geq 0. \tag{3.2}$$

Where $\|.\|_F$ denotes the Frobenius norm, β is the l_2 regularisation parameter, and the l_2 norm is defined as:

$$\|M\|_F^2 = \sum_{i=1}^{2k} \sum_{j=1}^{k} |m_{ij}|^2. \tag{3.3}$$

The first term in Eq. 3.2 is the fitting constraint on the real DDI data in X.

And the second term is the l_2 regularisation applied to M. Finally, I impose a non-negative constraint on M, as this favours biological interpretability because only additive combinations of the latent features will result in the final prediction [46].

SMH multiplicative algorithm

To minimise the loss function $\mathcal{J}_{\text{SMH}}(M)$ subject to the non-negative constraint, I developed a novel algorithm that uses the following multiplicative update rule:

$$m_{ij} \leftarrow m_{ij} \frac{(S^\top X H^\top)_{ij}}{(S^\top S M H H^\top + \beta M)_{ij}}. \tag{3.4}$$

This update rule is based on the principles of Non-negative Matrix Factorisation (NMF) and shares the same theoretical proofs of convergence [47]. Using this update rule to minimise Eq. 3.2 has some advantages over following gradient descent based approaches. There is no need to set a step-size parameter, it has theoretical guarantees of convergence, and it does not require a projection function to maintain the non-negative constraint. Following the update rule is enough to guarantee non-negativity, provided that M is initialised with non-negative values. M is initialised with random weights drawn from a uniform distribution in the interval $(0, \sqrt{\sigma})$.

To evaluate the performance of this approach, I remove 10% of the DDIs at random from X and hold them out for my testing set. I train a model with 80% of the DDIs, and the remaining 10% is used to select the optimal value for the l_2 regularisation parameter β. Once I find the optimal β, I train the final model with all the data available (90%) and evaluate the performance on the testing set. I make sure that all the rows in X have at least 5 ones in them, i.e. every drug pair is associated with at least 5 adverse side effects. This experiment is meant to evaluate how well the methods are able to predict novel DDIs for existing drugs on the market. This could potentially help clinicians who have to decide which drugs to co-administer and minimise the risk of adverse effects. A major drawback of this evaluation is that it is not fit to measure how a method performs at predicting DDIs for new drugs, since in this setup every drug pair needs to be have some known associations.

I use the Area Under the Receiver Operator Characteristic (AUROC) curve, and the Area Under the Precision-Recall Curve (AUPRC) to evaluate the performance of my approach. For both metrics, the perfect score is a value of 1. For the positive samples in training and testing, I use the DDIs from TWOSIDES, and the complement set of TWOSIDES is used as negative samples.

I obtain an AUROC of 0.702, and an AUPRC of 0.223 for the SMH model under this setting. This result indicates that although there is a minimal signal being captured by the matrix M, the performance can still be improved greatly. Some limitations of my approach could be in the constraints I imposed on the model. The first one is the non-negative constraint set on the matrix M. Another limitation of the model could lie in the learned representations that I fix for S and H. These low-dimensional vectors were learned using a different dataset and a different problem, predicting side effect frequency. Although this problem is related to DDI prediction, the fact that I am fixing this values and not allowing the model to learn the individual representations could also be limiting the performance of this approach. The last big constraint that I am setting onto the model, is learning M in a linear way.

Unconstrained SMH model

An important aspect of this approach is that it is trying to learn how two drugs interact with each other in order to cause an adverse side effect. Specifically, we can interpret these low-dimensional vectors as the distributed representations, of drugs and side effects, that encapsulate the state of their biological system. What the model tries to capture is how the drugs combine with each other to perturb different anatomical systems that will in turn cause the adverse side effect.

The SMH model has a non-negative constraint set on the matrix M. This is to favour the interpretability of the representations for the drug pairs. However, there might be a more complex relationship between two drugs that an additive combination may not be sufficient to model it. A DDI, normally, occurs when a drug affects the efficacy and safety of the other drug [14]. I hypothesise that removing the non-negative constraint on M, the performance of the SMH model can be improved. This will consider the possibility of having a trade-off between interpretability and predictive performance.

By removing the non-negative constraint imposed on M, I cannot longer use my algorithm that applies the multiplicative update rule to optimise $\mathcal{J}_{\text{SMH}}(M)$. Instead, I adopt a gradient descent approach to optimise Eq. 3.2. However, note that $\mathcal{J}_{\text{SMH}}(M)$ is differentiable and the differentiation can be solved for M. This means that I can use the closed form to obtain the optimal M by setting $\frac{\partial \mathcal{J}_{\text{SMH}}}{\partial M} = 0$:

$$\begin{aligned} \frac{\partial \mathcal{J}_{\text{SMH}}}{\partial M} &= -S^\top X H^\top + S^\top S M H H^\top \\ M &= (S^\top S)^{-1} S \top X H^\top (H H^\top)^{-1} \end{aligned} \tag{3.5}$$

Notice that normally the inverse of $(S^\top S)^{-1}$ and $(HH^\top)^{-1}$ would be considered costly operations. However, because the sizes of the matrices are small: $2k \times 2k$ and $k \times k$, respectively, with $k = 10$, these operations are not computationally intensive. I call this model unconstrained SMH (uSMH), to differentiate it from the SMH model.

To evaluate the performance of uSMH, I follow the same experimental setup described before. Table 3.1 shows the results compared to the traditional SMH model. As we can observe, the performance increases slightly for both the AUROC and the AUPRC. This indicates that removing the non-negative constraint may play a beneficial role in predictive performance. Inspecting the unconstrained M reveals that around 30% of the values in M go negative after removing the constraint. Nevertheless, the results are still not great. Following these results, the next thing to explore is to not use the learned low-dimensional vectors learned by Galeano et al. to represent the drugs and side effects. Another possibility is that the linearity of the model is limiting its capability to explain the interactions between the two drugs. These limitations will be explored in the next sections.

Table 3.1: Performance comparison between the constrained SMH and the unconstrained SMH models. For uSMH, M is allowed to have negative values.

Method	**AUROC**	**AUPRC**
SMH	0.702	0.223
uSMH	**0.713**	**0.240**

3.3 Non-linear model - MLP combination

The last preliminary model I present, explores the linear limitation of the SMH model. From the preliminary results of the SMH and uSMH models, I conclude that a linear combination may not be sufficient to capture the intricacies involved in the combination of two drugs. Deep neural networks can use many intermediate layers of artificial *neurons* between the input and output to learn a hierarchy of progressively more complex feature detectors [48]. For this approach, I make use of the drug and side effect distributed representations, learned by Galeano et al., but instead of learning a linear combination of the individual drug representations, I use a Multi-Layer Perceptron (MLP). This means connecting a deep neural network to the individual drug representations to model the non-linear association between them, and learn a representation for the drug pair. This representation for the drug pair will be connected to the representations of the side effects to make a prediction for the association.

Figure 3.2 shows a schematic of this approach. The input for the model are the representations of the individual drugs. These representations are fixed and are not learned by my method. They are denoted by the matrix W learned by the method by Galeano et al. The row i of W represents the low-dimensional vector for drug i. These representations w_i and w_j are then fed to the MLP combination layers to learn the non-linear combination between them:

$$\vec{c}_{i,j} = \phi_{out}\left(\phi_L\left(\ldots\phi_2\left(\phi_1\left(\vec{w}_i \oplus \vec{w}_j\right)\right)\ldots\right)\right), \tag{3.6}$$

here $\oplus$ represents the concatenation, ϕ_{out} and ϕ_x respectively denote the mapping function for the output layer and l_{th} hidden layer, where there are L hidden layers in total, and $\vec{c}_{i,j} \in \mathbb{R}^{k\times 1}$ signifies the distributed representation for the combination of drugs i and j.

Finally, the vector of predicted scores for the drug pair (i, j), $\hat{x}_{i,j}$, where $\hat{x} \in \mathbb{R}^{1\times m}$, with $m = 544$, is obtained by:

$$\hat{\vec{x}}_{i,j} = \vec{c}_{i,j}H \tag{3.7}$$

The output layer of this approach is a non-trainable layer where the weights are fixed by the matrix H containing the side effects representations learned by Galeano et al. This approach is an extension of the uSMH model, where instead of learning M linearly, an MLP is used to model the nonlinearities involved in the combination of two drugs. I believe that the use of nonlinearity will help capture a more meaningful relationship between the drug representations.

Table 3.2: Performance comparison between all the preliminary models.

Method	**AUROC**	**AUPRC**
SMH	0.702	0.223
uSMH	0.713	0.240
MLP combination	**0.785**	**0.297**

Table 3.2 includes the results of this approach compared against the other preliminary models. As observed, the MLP combination approach yields much better results than the uSMH. This indicates that the linearity was limiting the modelling of the combination of the two drugs, and may hint that the drugs are actually combining nonlinearly.

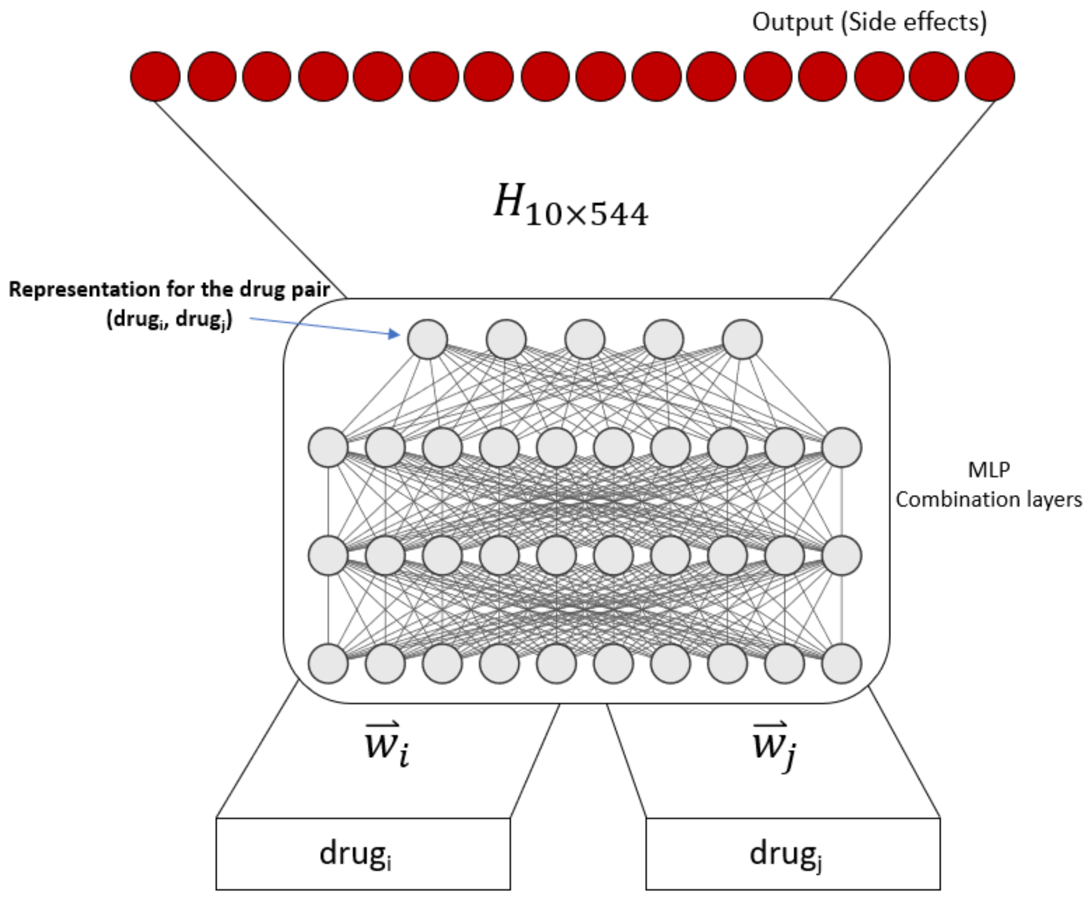


Figure 3.2: MLP combination model. This method learns a non-linear combination for a drug pair, based on the two fixed learned representations of the individual drugs, represented by w_i and w_j. The output of the MLP is the representation of the drug pair, which gets multiplied by the representations of the side effects to obtain a predicted score for each side effect in the output layer.

3.4 Final remarks

In this chapter, I presented the early models that paved the way to my final approach that is presented in the next chapter. The main idea behind these models is to learn how to combine the distributed representations of two individual drugs to learn a representation for a drug pair. These representations, for drugs and side effects, found a lot of success at predicting the frequency of single drug side effects [44]. For this reason, finding the right way to combine, or learn, them was an important objective of these initial approaches.

The main findings from the early models are detailed as follows:

- A linear approach is not good enough to model how two drugs combine in order to cause new adverse side effects.
- Using an MLP to learn a representation for the drug pairs, shows how much the approach benefits from nonlinearity. There are a lot of complex interaction between the individual representations that cannot be captured by a simple linear approach.

Another important remark is that these methods are very rigid in terms of the experimental setups that can be adopted for evaluation. The current literature of DDI prediction adopts different settings, which can't be adapted for most of the preliminary methods showed here. Nevertheless, these results are good to show that the learning is going in the right direction. In Chapter 5, I provide a detailed comparison of my final approach against state-of-the-art methods in DDI prediction under different evaluation protocols.

My final method incorporates the strength of each of the preliminary models, while also addressing their limitations. It learns distributed representations directly from the known information, without relying on external pre-trained data. My approach also makes use of data of single drug side effect to learn more robust representations for both drugs and side effects. It is highly nonlinear, as it uses an MLP to model the complex relationships between the individual drugs. In the next chapter, I explain this approach in full detail.

"All the knowledge is in the connections."

David Rumelhart

4 Learning distributed representations for predicting adverse effects of drug combinations

In this chapter, I present DCSE-twin, a novel approach for predicting adverse side effects caused by DDIs. The idea behind my approach is to learn distributed representations separately for drugs and side effects. These are learned in a neural network that combines, nonlinearly, the representations of the individual drugs to obtain the representation for drug pairs. The elements in the distributed representations, can be interpreted as the activations of different biological systems. When my approach relates the representation for a drug pair, with the representation of a side effect, it obtains a score that can be interpreted as the probability of the side effect being associated to the drug pair. The fact that these representations are explicitly separate, allows DCSE-twin to synergistically learn from side effect data of both single drugs and drug pairs. This provides my model, the ability to answer novel questions, in the field of DDI prediction, that are relevant for pharmacology. DCSE-twin can address the *cold start* problem by predicting adverse DDIs, for drug pairs for which no side effects are known, or even for drug pairs in which neither of the drugs is known to interact with any other drug.

An overview of DCSE-twin is presented in Figure 4.1. The method is a combination of two models, DCSE-left and DCSE-right. DCSE-left is a non-negative matrix factorisation model that utilises data from side effects caused by single drugs. DCSE-right is a deep neural network that learns from side effects caused by drug-drug interactions. The two architectures in this approach are linked, as the weights for the distributed representations are shared between the two models. By sharing the weights across the models, the representations can be learned from two different data sources simultaneously. DCSE-left allows the approach to learn representations that are more descriptive of the mechanism of single drug action and lets the neural network, in DCSE-right, model the nonlinearities involved in the combination of two drugs. As the weights are updated by one of the models to incorporate new knowledge about drugs or side effects, the changes affect the knowledge associated with other entities in the other model that are represented by similar activity patterns.

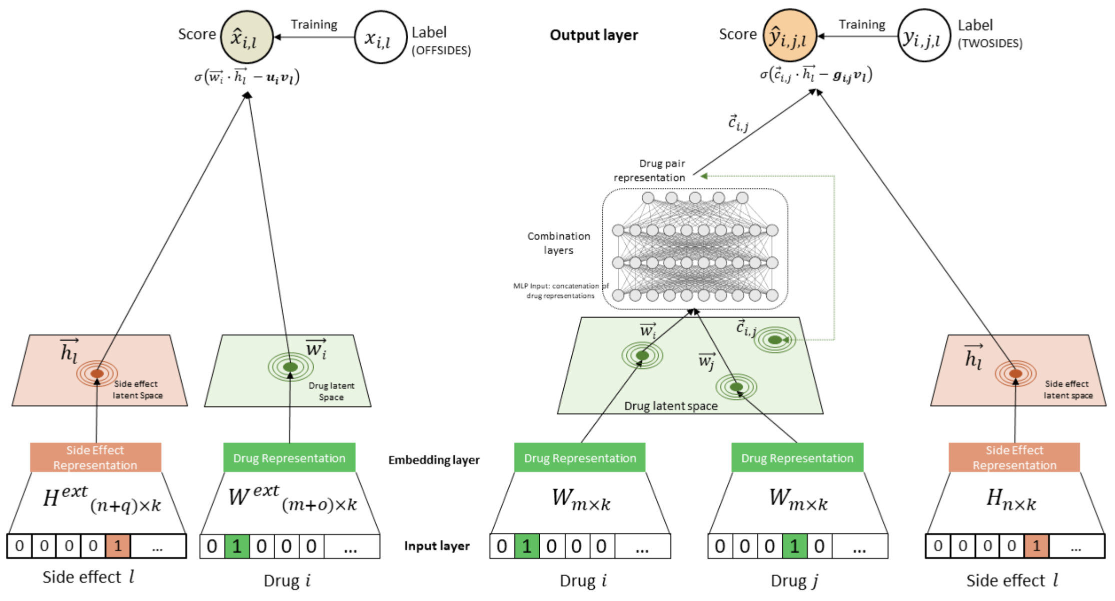


FIGURE 4.1: Overview of the architecture of my final model, DCSE-twin. It consists of the combination of DCSE-left and DCSE-right. The two models share the parameters for the distributed representations of drugs and side effects. The final prediction for my approach is given by the output of DCSE-right, $\hat{y}_{i,j,l}$, and it represents the probability that a side effect l is caused by the co-administration of drugs i and j.

A key distinction between the early methods showcased in Chapter 3 and this final approach, is the way in which the problem is framed. Motivated by recent work in collaborative filtering [49], I restructure the architecture of the model by replacing the n side effect outputs, to a single output that determines the probability of a combination of drugs being associated to a given side effect. In this new architecture, the side effect is also an input to the model. This change allows the approach to be more robust and flexible in terms of the different experimental settings and research questions that can be tackled, all of which are explained in Chapter 5.1. It also means that the number of side effects that the model can predict is not limited, nor restricted by the architecture. As long as we have the distributed representation for drugs and side effects, my model will be able to predict a probability for the drug-drug interaction.

An important characteristic of DCSE-twin is that the distributed representations are constrained to be non-negative. This has the added advantage of making explicit the parts-based representation [47], and therefore, having the potential to offer biological interpretability. This means that drugs are characterised by a set of learned non-negative low-dimensional features that, when additively combined with the representation of the side effects, account for the probability of that side effect occurring or not. From the work by Galeano et al., we learned that these representations can then be used to explain drug activity in terms of perturbations of distinct biological and anatomical systems. By also constraining the learned representations for the drug combinations to be non-negative, I expect the same notion of interpretability can also be extended for drug-drug interactions.

In this chapter, I will first focus on providing the mathematical formulations for the two models in the approach, DCSE-right and DCSE-left. Finally, I will detail the final approach, DCSE-twin, that is the combination of the two separate models.

4.1 DCSE-right, predicting adverse DDIs

DCSE-right has the lead role of predicting the side effects caused by drug-drug interactions. It considers the task of identifying associations between drug pairs and side effects. Let D be a set of drugs and R a set of side effects, then the drug-drug interaction prediction task can be considered as a function $f : D \times D \times R \to [0, 1]$. For a given triplet of two drugs and a side effect (d_i, d_j, r_l), the aim is to determine how likely it is that the concurrent use of drugs d_i and d_j is associated with the side effect r_l in the human patient population. The goal is to find an approximation of f, given a dataset of known associations between drug pairs and side effects $T = \{(d_i, d_j, r_l)_p | d_i, d_j \in D, i \neq j, r_l \in R\}_{p=1}^{P} \subset D \times D \times R$, where P is the total number of known triplets. The model described below is one such approximation.

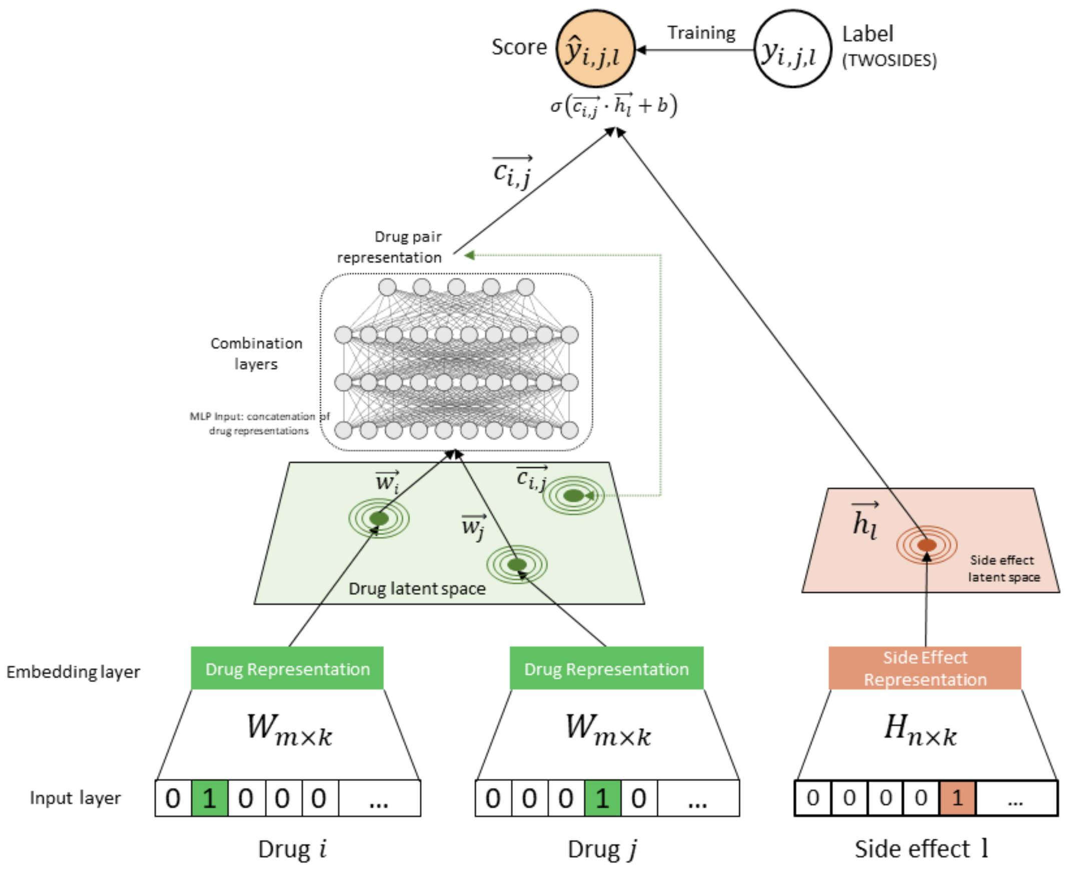


Figure 4.2: DCSE-right. The individual representations for the two drugs are fed into a neural network. These combination layers will provide the necessary nonlinearities to find a representation for the drug pair. The final prediction score is given by $\hat{y}_{i,j,l}$. This model is trained with data from TWOSIDES.

The architecture of the model can be observed in Figure 4.2. The bottom input layer consists of three vectors $\vec{d_i}, \vec{d_j}$ and $\vec{r_l}$ with a one-hot encoding that determines the identity of the two drugs and the side effect, respectively. Above the input layer, is the embedding layer, a fully connected layer that projects the sparse encoding to a dense vector. Note that to maintain the same representations for all drugs, I use a Siamese embedding layer $W \in \mathbb{R}^{m\times k}$, where m is the number of drugs in T and k is the embedding size. For the side effect input, the embedding layer is $H \in \mathbb{R}^{n\times k}$, where n is the number of side effects in T. Note that $k \ll \min(m, n)$ and represents the number of latent features in the distributed representations. Thus, the representations for the drugs and side effects are obtained through the following:

$$\begin{aligned} \vec{w_i} &= W^T \vec{d_i} \\ \vec{h_l} &= H^T \vec{r_l} \end{aligned} \tag{4.1}$$

where $\vec{d} \in \mathbb{R}^{m\times 1}, \vec{r} \in \mathbb{R}^{n\times 1}$ and $\vec{w}, \vec{h} \in \mathbb{R}^{k\times 1}$. The resulting drug and side effect embeddings are the distributed representations learned by DCSE-right.

The representations for the two drugs are then concatenated and fed into a multi-layer neural architecture, which we term the combination layers. The output of the combination layers has the same size as the distributed representations for the drugs and it signifies the learned distributed representation for the drug pair. I define the combination layers as follows:

$$\vec{c}_{i,j} = \phi_{out}\left(\phi_X\left(\ldots\phi_2\left(\phi_1\left(\vec{w_i} \oplus \vec{w_j}\right)\right)\ldots\right)\right), \tag{4.2}$$

here $\oplus$ represents the concatenation, ϕ_{out} and ϕ_x respectively denote the mapping function for the output layer and x_{th} hidden layer, where there are X hidden layers in total, and $\vec{c}_{i,j} \in \mathbb{R}^{k\times 1}$ signifies the distributed representation for the combination of drugs i and j. These hidden layers are meant to provide a large level of flexibility to model the nonlinear interactions between the latent features of the representations of the two drugs. The fundamental assumption is that the hidden layers might also capture some specific molecular mechanisms between the two drugs that make them interact in a way that would elicit side effects.

Finally, the predicted score $\hat{y}_{i,j,l}$ is obtained by

$$\hat{y}_{i,j,l} = \sigma\left(\vec{c}_{i,j}^{\,\mathrm{T}}\vec{h}_l + b\right) \tag{4.3}$$

The term $\vec{c}_{i,j}^{\,T}\vec{h}_l$, is the dot product between the representations of the drug pair and the side effect, and b is the bias term. σ represents the sigmoid activation function. The target value $y_{i,j,l}$ is interpreted as a probability between 0 or 1 indicating whether the side effect l is associated to the combination of drugs i and j.

The bias term b plays a key role in the prediction. Its first purpose is to make sure that the entire range of the sigmoid function is being represented. To understand why this is the case, notice that since both $\vec{c}_{i,j}$ and $\vec{h}_l$ are non-negative, the lower bound for the dot product between these two vectors is 0. This means that if we omit the bias term, and considering the nature of the sigmoid function[1], our prediction $\hat{y}$ will be bounded between 0.5 and 1 . The bias term ensures that the raw value of the non-normalised predictions is allowed to go below 0. Consequently, the final prediction values fall within the entire range of the sigmoidal function. The second purpose of the bias term is to model the systematic tendencies inherent to the data, such as, the prescription bias. Some drugs are, generally, prescribed more than others, and some side effects tend to occur more frequently than others [50]. Thus, it would be unwise to explain the total prediction score with just the dot

[1] $\sigma(a) = \frac{1}{1+\exp(-a)}$

product of the representations. Instead, the system tries to identify the portion of the prediction that can be attributed to the bias, subjecting the true interaction portion of the data to the modelling of the distributed representations.

The deep neural network in DCSE-right integrates the problem-specific knowledge into the learning. It imposes constraints on how the distributed representations of the drugs and side effects are being integrated. The idea is that DCSE-twin favours interpretability of the distributed representations. As shown before in [44], the low-dimensional vectors can be analysed to gain a biological interpretation for the predictions. For this reason, the architecture in DCSE-twin is built so that it is possible to discern how the features of the drugs and side effects (that will be learned linearly) are combined (nonlinearly) to make the predictions.

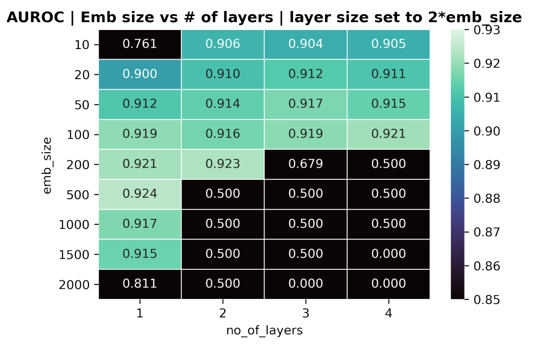


FIGURE 4.3: Heatmap of the hyperparameter grid search for the combination layers - Values represent the AUROC.

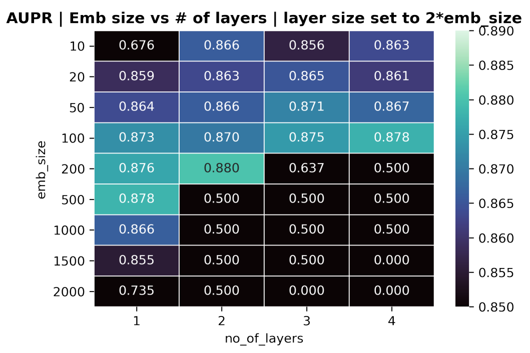


FIGURE 4.4: Heatmap of the hyperparameter grid search for the combination layers - Values represent the AUPRC.

For the implementation of the distributed representations of single drugs and side effects, W and H are initialised by sampling from a uniform distribution in the range $[0, 0.01]$, as described in [44]. The combination layers have the same dimensionality as the concatenation of the representations of the two individual drugs, i.e., $2 \times k$. As for the number of hidden layers and the size of the representations, I adopt a grid search to find an appropriate value for these hyperparameters[2] (as seen in Figures 4.3 and 4.4). For the number of hidden layers, I select 2. For the size of the representations I select 200 for the experiments shown in Chapter 5, as it achieves the higher predictive performance. However, when considering the interpretability of the distributed representations, I select 20 as the size, as a smaller size would allow a more comprehensive interpretation. Note that after extensive testing, the results show that DCSE-twin, with this smaller size for the representations, still outperforms state-of-the-art methods, as this score (0.914) is higher than the next-top competitor shown in Table 5.2. A major advantage of DCSE-twin is its robustness to the setting of the parameters. As long as some reasonable values are selected, the performance does not vary significantly, and I can select parameters that can, for instance, favour interpretability, without losing predictive performance.

Finally, after the last hidden layer, we obtain the representation for the pair, $\vec{c}_{i,j}$, with the same size as the vectors for the single drugs, i.e., k. The distributed representations for the drug pair also lie in the same latent vector space as the representations of the single drugs. This characteristic is crucial to keep the model interpretable, and also scalable in terms of the number of drugs in the combinations. Even if there are more drugs in the combination, DCSE-right will learn a nonlinear combination to represent the combination of drugs in the same latent space as the individual drugs.

For the choice of the nonlinear activation function, I first use the hyperbolic tangent, tanh, as the activation for the first hidden layer that initially combines the two representations of the individual drugs. The idea behind using the tanh is to allow

[2]The performance of the grid search is evaluated on a validation set, independent from the testing set.

for negative activations as the first step to combining the two single representations. There may be some interactions between the drugs that may be better represented by negative values. Empirical results show little variation in the overall performance of the method by replacing the hyperbolic tangent with another activation function, such as, the Rectified Linear Unit (ReLU) or the sigmoid functions. For the rest of the hidden layers, and the drug pair representation, I select ReLU for the nonlinear activation function. Notice that the values of the representations for the single drugs are not upper-bounded, this makes ReLU a perfect choice for the activation of the drug pair representation, since the idea is that the distributed representations all lie in the same latent space. Also, ReLU encourages sparse activations which may play an important role in the interpretability of the latent vectors, as we expect only some components to be activated for specific drugs. My empirical results show that ReLU yields better performance than the tanh or sigmoid functions. The training time is also significantly reduced.

Note that the nonlinearity is given by the combination layers in the form of a MLP. However, there are alternatives that may yield a different combination. One such alternative is using a Convolutional Neural Network (CNN). In the Appendix A.1 I include a variation of DCSE-right which uses a CNN instead of an MLP and achieves similar performance.

4.1.1 Learning DCSE-right

Considering the binary nature of the drug-drug interaction data, we can view the value of $y_{i,j,l}$ as a label, where 1 means that the drug pair (i, j) is associated to the side effect l. The predicted score $\hat{y}_{i,j,l}$ then represents the probability of l being associated to the drug pair (i, j). My method then determines the model parameters by minimising the binary cross-entropy loss function, also known as the negative log loss:

$$\min_{W,H,\theta_c,b} \mathcal{L}\left(W,H,\theta_c,b\right) = -\sum_{i,j,l} y_{i,j,l} \log \hat{y}_{i,j,l} + \left(1-y_{i,j,l}\right) \log\left(1-\hat{y}_{i,j,l}\right) \tag{4.4}$$
$$\text{subject to } W,H \geq 0$$

Where θ_c represents the parameters of the hidden combination layers. The model is estimated through negative sampling as described in [26]. For each (d_i, d_j, r) triplet (i.e., a positive example), I sample a triplet (d_i, d_n, r) (i.e., a negative example) by randomly selecting drug d_n according to a sampling distribution P_r [26]. Further explanation of the negative sampling procedure can be found in Chapter 5.1.2. As described in the work by Zitnik et al. [1], I maintain a balanced ratio of positive to negative examples in the training set. To train the model I use the Adaptive Moment Estimation (Adam) [51], which adapts the learning rate for each parameter by performing smaller updates for frequent and larger updates for infrequent parameters. Adam results in faster convergence and higher performance in general than vanilla Stochastic Gradient Descent (SGD), as my empirical results indicate. The code for DCSE-right was implemented in Python 3.9.7 and the TensorFlow 2.3.1 library.

4.2 DCSE-left, predicting off-label side effects for single drugs

The next important component of DCSE-twin is the left-side model in my approach, DCSE-left. It consists of a regularised logistic NMF model that learns representations of drugs and side effects. The main difference between the two models in my approach is that DCSE-left learns from single drugs - side effect data and DCSE-right learns from pairs of drugs - side effect data. The main motivation to use this model is to augment the amount of data that we can use to learn the distributed representations of drugs and side effects. This allows DCSE-twin to improve its overall predictive performance, and to answer new biologically relevant research questions. The idea is to leverage this data to improve the learning of the representations. Nevertheless, there is another reason why using DCSE-left benefits the overall approach. This model does not only learn the distributed representations for the same drugs and side effects as DCSE-right but also learns them for many that don't have any drug-drug interaction information. This means that by plugging the representations, for these drugs and side effects, in DCSE-right, we can obtain a prediction for drug pairs and side effects with previously no recorded associations. This setting is called *de novo* prediction. It allows us to overcome the famous *cold-start* problem where a prediction can't be made unless there is some previous information about the entities involved. In Chapter 5.1.5 I include a detailed explanation of all the different scenarios for which I can obtain these *de novo* predictions, along with the experiments carried out.

Let's again start by formulating the problem I am tackling with DCSE-left. This problem considers the task of identifying associations between single drugs and off-label side effects. Let D^{ext} be a set of drugs and R^{ext} a set of side effects, where $|D^{ext}| = m + o$, $|R^{ext}| = n + q$, and o and q are the number of drugs and side effects, respectively, available only for the left-side data. Then for a drug - side effect (d_i, r_l) tuple, the aim is to determine how likely it is that the use of drug d_i is associated to the side effect r_l in the human patient population. Let Z be the set with the known associations between drugs and off-label side effects, where $Z = \{(d_i, r_l)_j | d_i \in D^{ext}, r_l \in R^{ext}\}_{j=1}^{J} \subset D^{ext} \times R^{ext}$, and J is the total number of known tuples. Then we can define a binary $(m+o) \times (n+q)$ matrix X, with $(m+o)$ unique drugs and $(n+q)$ unique side effects. Every association in Z is represented as a 1 in X, and the remaining entries of the matrix are filled with zeros.

Therefore, the idea is to learn a low-dimensional distributed representations for drugs, defined as $\vec{w} \in \mathbf{R}^k$, and for side effects, defined as $\vec{h} \in \mathbf{R}^k$. Such that, the score for a drug - side effect association, $\hat{x}_{i,l}$, is given by the dot product of the two distributed representations:

$$\hat{x}_{i,l} = \vec{w}_i \vec{h}_l \tag{4.5}$$

This score can be obtained by decomposing X into a product of two matrices as $X \simeq W^{ext} H^{ext}$, where $W^{ext} \in \mathbb{R}^{(m+o) \times k}$ (each row is a distributed representation for a drug), and $H^{ext} \in \mathbb{R}^{k \times (n+q)}$ (each column is a distributed representation for a side effect). W^{ext} and H^{ext} can be learned by optimising the Least Squares (LS) loss function, either by gradient descent or using Multiplicative Update Rules (MURs) as described in [47, 52]. However, learning from LS maps onto the entire positive real space of numbers. This is not the ideal scenario for this task, since the target data, X, is binary, and the predictions won't be upper-bounded to 1.

4.2.1 Logistic NMF

DCSE-left plays an important role in my approach, so it is crucial to incorporate it with DCSE-right in the proper way. For binary data, a general NMF is not optimal and it may fall behind in performance when compared to probabilistic approaches [53]. Inspired by the work by Larsen and Clemmensen [52], I developed the Logistic NMF model (LogNMF) for the prediction of side effects caused by single drugs. This LogNMF is a generative model formed by the combination of NMF and logistic regression. The first change to the traditional NMF would be to the prediction score, $\hat{x}_{i,l}$, in Eq. 4.5:

$$\hat{x}_{i,l} = \sigma(\vec{w}_i \vec{h}_l - b). \tag{4.6}$$

Two things are added here, the sigmoid function σ, and the bias term b. The bias has the same functions as described in the previous section for DCSE-right. Note that for the prediction in LogNMF (Eq. 4.6), the bias term has a minus sign. This is because the same non-negative constraints applied to the matrices W^{ext} and H^{ext}, also apply to the bias b. This bias term can be defined, in its general form, as a global constant value, b that applies to every $\hat{x}_{i,l}$. However, I define it as the product of two different biases, $u_i v_l$, where one value represents the bias for the drug i and the other the bias for side effect l. These biases play the role of capturing the general trends in the data (e.g. drug and side effect popularity, see Chapter 5.4), allowing the distributed representations to learn from more biologically relevant associations.

Let's define $\vec{u} \in \mathbf{R}^{(m+o)\times 1}$ as the vector with the values for the biases for each individual drug, and $\vec{v} \in \mathbf{R}^{(n+q)\times 1}$ as the vector with the values for the biases for each side effect. This allows having a specific bias value for each drug and for each side effect separately, and consequently, for each $\hat{x}_{i,l}$. Considering this, the final prediction for LogNMF has the following form:

$$\hat{x}_{i,l} = \sigma(\vec{w}_i \vec{h}_l - u_i v_l). \tag{4.7}$$

A representation of DCSE-left can be observed in Figure 4.5. The input layer consists of two binarised sparse vectors $\vec{d_i}$ and $\vec{r_l}$ with a one-hot encoding that determines the identity of the drug and the side effect, respectively. Above the input layer, is the embedding layer which contains the values for the representations of both drugs and side effects, depicted by W^{ext} and H^{ext}. To obtain the latent vector for the drug and the side effect, namely, $\vec{w_i}$ and $\vec{h_l}$, the same operation as in Equation 4.1 is carried out. Lastly, the output consists of the predicted probability of drug i causing side effect l, given by Eq. 4.7.

To learn the representations and the biases, I minimise the regularised LogNMF loss function:

$$\begin{gathered}\min_{W^{ext},H^{ext},u,v} \mathcal{L}_{LogNMF}\left(W^{ext}, H^{ext}, b\right) = -\sum_{i,l} x_{i,l} \log \hat{x}_{i,l} + (1 - x_{i,l}) \log (1 - \hat{x}_{i,l}) \\ +\frac{\beta}{2}\|W^{ext}\|_F^2 + \frac{\beta}{2}\|H^{ext}\|_F^2 + \frac{\alpha}{2}\left\|uv^T\right\|_F^2 \\ \text{subject to } W^{ext}, H^{ext}, u, v \geq 0.\end{gathered} \tag{4.8}$$

Here $\|.\|_F$ denotes the Frobenious norm, and β and α are regularisation parameters. The goal is to learn meaningful low-dimensional representations that in a way

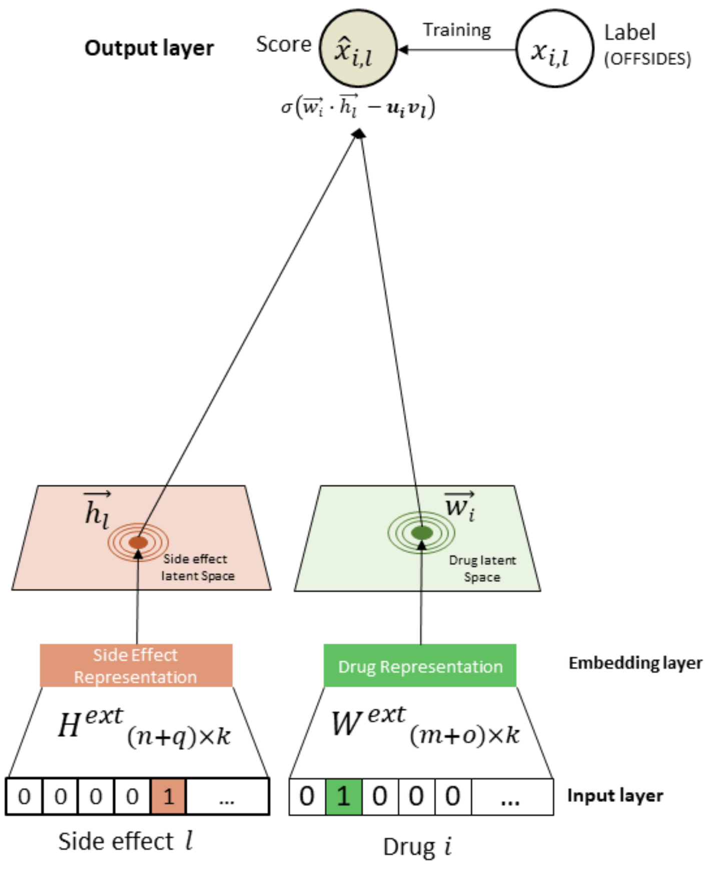


FIGURE 4.5: DCSE-left: LogNMF. A logistic variation of the classic NMF to learn from binary data. The prediction is constrained to $[0, 1]$, but the low-dimensional distributed representations are kept non-negative. This model is trained with data from off-label side effects caused by single drugs — OFFSIDES.

they are able to generalise and capture the biological activations of drugs and side effects. Thus, the model should avoid overfitting the observed data by penalising the magnitudes of the learned parameters. The constants β and α control the extent of the regularisation for the distributed representations and the biases, respectively. The values for these hyperparameters are determined using a validation set (see Chapter A.2.2).

I was able to derive efficient multiplicative learning algorithms that do not require setting a learning rate or using a projection function to guarantee non-negative constraints. I will show now the derivation of this rule for W^{ext}[3]. Similarly to Lee and Seung [47], I start by applying the gradient descent update rule on Eq. 4.8:

$$\begin{aligned} W &\leftarrow W - \eta \nabla \mathcal{L}_{LogNMF}(W) \\ W &\leftarrow W - \eta\left[XH^T - \left(\hat{X}H^T + \beta W\right)\right] \end{aligned} \tag{4.9}$$

where $\nabla(.)$ represents the derivative with respect to W, and $\hat{X} = \sigma(W^{ext}H^{ext} - uv^T)$ denotes the matrix with all the predicted values.

If $\eta = c$, for a constant value c, then the algorithm in Equation 4.9 is equivalent to gradient descent with a constant step size. Instead if I re-scale the variables and set:

$$\eta \text{ to } \rightarrow \frac{W}{\hat{X}H^T + \beta W} \tag{4.10}$$

then, Equation 4.9 becomes:

$$W \leftarrow W + W \circ \frac{XH^T}{\hat{X}H^T + \beta W} - W \circ \frac{\hat{X}H^T + \beta W}{\hat{X}H^T + \beta W} \tag{4.11}$$

Where $\circ$ represents the Hadamard, or element-wise product. It is straightforward to see that by cancelling out some terms in Equation 4.11, I can obtain the following multiplicative learning rule for W:

$$W \leftarrow W \circ \frac{XH^T}{\hat{X}H^T + \beta W} \tag{4.12}$$

[3]Note that to avoid a convoluted notation, the superscript ^{ext} was omitted from the matrices W and H.

A similar procedure is followed for H, u and v, to obtain the following multiplicative rules:

$$\begin{aligned} H &\leftarrow H \circ \frac{W^T X}{W^T \hat{X} + \beta H} \\ u &\leftarrow u \circ \frac{\hat{X} v}{X v + (\alpha v^T v)\, u} \\ v^T &\leftarrow v^T \circ \frac{u^T \hat{X}}{u^T X + (\alpha u^T u)\, v^T} \end{aligned} \tag{4.13}$$

These update rules themselves ensure that the non-negative constraint is satisfied. This is made possible by the re-scaling of the learning rate for every entry in W using Equation 4.9. Note that the model parameters can also be learnt using stochastic gradient descent and a projection function to guarantee the non-negative constraints. However, this procedure does not require setting a learning rate nor applying such projection function. Another reason why I decided to use the MURs, instead of gradient descent, is to stabilise the changes in the model parameters between iterations (see Chapter 5.5). This should potentially help the reproducibility of the model and, following that, the interpretability of the distributed representations.

4.3 DCSE-twin

The final model of my approach consists of the combination of DCSE-left and DCSE-right, previously described in this chapter. We return to the overview in Figure 4.1. These two models will learn, simultaneously, the distributed representations for drugs, pairs of drugs and side effects. The idea is to have both models benefit from each other by improving the representations of drugs and side effects by incorporating the extra amount of data each model brings. In order for the representations to leverage the data from both problems, the learning for the two models is done by taking turns. And, the key aspect is that the weights of the embedding layers for drugs and side effects, i.e., the distributed representations, are transferred to the other model in between turns in the learning. This also allows the problem-specific information of each model to be transferred to the other one. Drugs or side effects that are being updated in one model, will affect the knowledge of drugs and side effects that have similar activity patterns in their representations.

Besides the distributed representations of drugs and side effects, I am also transferring the biases. Sharing the bias between the two models allows the system to capture the drugs' and side effects' tendencies and noise in the data for both different problems. Thus, the model can leverage the actual interaction signal in the data for learning the distributed representations.

To allow the sharing of the biases between the two models, a few modifications need to be made to DCSE-right. First, let $u \in \mathbf{R}^{m\times 1}$ and $v \in \mathbf{R}^{n\times 1}$ be the drug and side effect bias vector, respectively. Let us refer to the bias vectors in DCSE-left as $u^{ext} \in \mathbf{R}^{(m+o)\times 1}$ and $v^{ext} \in \mathbf{R}^{(n+q)\times 1}$ to differentiate between the biases of the two models. Now we can modify the final prediction of DCSE-right in Equation 4.3, as follows:

$$\hat{y}_{i,j,l} = \sigma\left(\vec{c}_{i,j}^{\,\mathrm{T}}\vec{h}_l - g_{i,j}v_l\right) \tag{4.14}$$

Here, $g_{i,j}$ represents the bias for the drug pair (i, j). Remember that for DCSE-left, the bias value for the drug-side effect pair is given by the product of the bias of the drug and the side effect: $u_i^{ext} v_l^{ext}$. For DCSE-right, I explicitly model the bias for the drug pair $g_{i,j}$, and argue that the combination of the two individual biases, u_i and u_j, needs to also be nonlinear. I will use an MLP to model the combination bias for two drugs. A graphic representation of this notion is depicted in Figure 4.6.

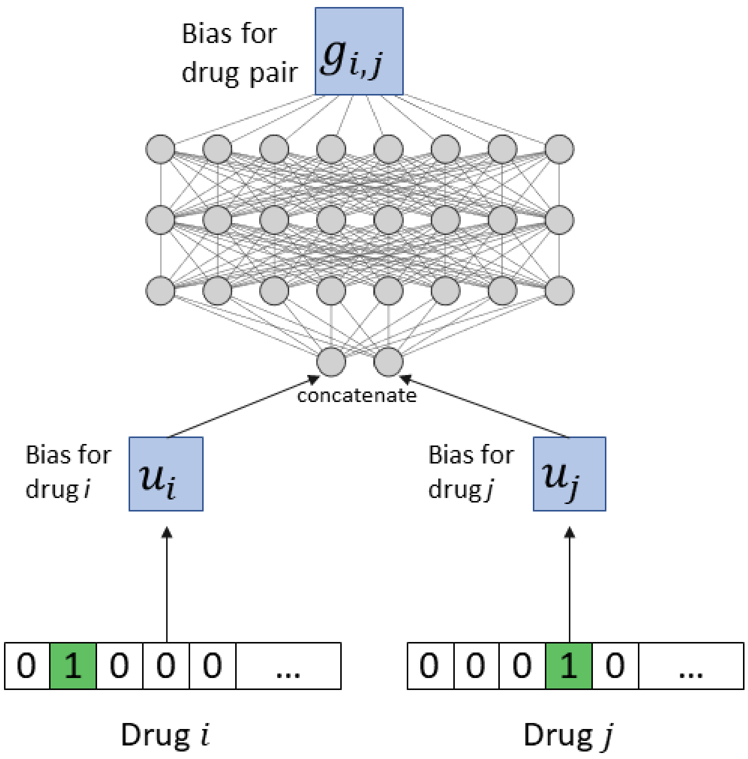


FIGURE 4.6: Explicit modelling of the combination bias. An MLP is used to learn how the individual drug biases combine to come up with a single value that represents the bias for the drug pair.

Let us then define $g_{i,j}$, in a similar way as Eq. 4.2, as follows:

$$g_{i,j} = \phi_{out}\left(\phi_B\left(\dots \phi_2\left(\phi_1\left(u_i \cdot u_j\right)\right)\dots\right)\right), \tag{4.15}$$

where $\cdot$ represents the concatenation, ϕ_{out} and ϕ_x denote the mapping function for the output layer and b_{th} hidden layer, respectively. And, there are B hidden layers in total. Finally, the loss function for DCSE-right is also minimised for the parameters of the bias combination MLP, θ_g:

$$\begin{gathered}\min_{W,H,\theta_c,\theta_g} \mathcal{L}\left(W,H,\theta_c,\theta_g\right) = -\sum_{i,j,l} y_{i,j,l} \log \hat{y}_{i,j,l} + \left(1-y_{i,j,l}\right) \log\left(1-\hat{y}_{i,j,l}\right) \\ \text{subject to } W,H,u,v,g \geq 0.\end{gathered} \tag{4.16}$$

4.3.1 Training DCSE-twin

The training of DCSE-twin, is done by taking turns between the two models. This allows to transfer the knowledge of one problem to the other, and learn more meaningful representations for drugs, drug pairs and side effects.

The pseudo-code of this algorithm is described in Algorithm 3. First, the training starts by initialising the model parameters. Followed with a few initial iterations (`initialIters`) of the algorithm of DCSE-left to give the representations a good initial value. After these iterations, the main training loop begins. Before moving the learning to DCSE-right, the weights from DCSE-left are transferred to DCSE-right. Note that the number of drugs and side effects in DCSE-left is larger than for the right side. So, only the weights for the common drugs and side effects are being shared. Then, the training moves to DCSE-right, where a number of epochs are run (`turnEpochs`). After transferring the learned weights from the right-side to DCSE-left, the weights get updated again by DCSE-left a certain amount of times (`turnIters`). After a fixed total amount of turns (`totalTurns`), one last update is done on DCSE-right since this is the prediction task we care about.

The different variables in Algorithm 3, namely, `initialIters`, `totalTurns`,

ALGORITHM 3: Training algorithm for the final model.

```
1 $W^{ext}, H^{ext}, u^{ext}, v^{ext} \leftarrow$ initialiseWeights();
2 $W^{ext}, H^{ext}, u^{ext}, v^{ext} \leftarrow$ leftSideTraining(iters = initialIters);
3 for $i \leftarrow 1$ to $totalTurns$ do
4     $W, H, u, v \leftarrow W^{ext}[: m, :], H^{ext}[: n, :], u^{ext}[: m], v^{ext}[: n]$;
5     $W, H, u, v \leftarrow$ rightSideTraining(epochs = turnEpochs);
6     $W^{ext}[: m, :], H^{ext}[: n, :], u^{ext}[: m], v^{ext}[: n] \leftarrow W, H, u, v$;
7     $W^{ext}, H^{ext}, u^{ext}, v^{ext} \leftarrow$ leftSideTraining(iters = turnIters);
8 $W, H, u, v \leftarrow W^{ext}[: m, :], H^{ext}[: n, :], u^{ext}[: m], v^{ext}[: n]$;
9 $W, H, u, v \leftarrow$ rightSideTraining(epochs = lastEpochs);
```

`turnEpochs`, and `turnIters`, are set empirically after a thorough analysis of the learning dynamics between the two models, described in Chapter A.2.

"The scientist is not a person who gives the right answers, he's one who asks the right questions."

Claude Levi-Strauss

5 Experimental Results

In this chapter, I present the results I obtained by extensively testing DCSE-twin in a variety of experimental settings and comparing it with state-of-the-art methods. I also introduce several settings that I developed in an attempt to approximate real-world scenarios in drug development, pharmacovigilance and drug repurposing. The results show that DCSE-twin outperforms the existing state-of-the-art method in every experimental setting. Moreover, these experiments allowed me to shed some light on the biological aspects of the problem as well as on the inner workings of my algorithm.

The experimental settings and the research questions they attempt to answer are the following:

- **Setting A.** The research question is: *how good is a method at predicting drug pairs that might cause a specific side effect?* This is a standard evaluation procedure that appears in the existing literature and was introduced by Zitnik et al [1].

- **Setting B – *De novo* prediction of side effects for unseen drug pairs.** This setting answers the question: *how good is a method at predicting the potential side effects for a drug pair with no known interactions in the DDI dataset (that I use for training)?*

- **Prospective evaluation.** Here the question is: *based only on past information, how good is an approach at generalising its predictions for newly discovered adverse DDIs?* To the best of my knowledge, my work is the first to perform a prospective evaluation for adverse DDI prediction. For this evaluation, I train the models on an older version of TWOSIDES that includes data up to 2009 and use them to predict new associations reported from 2010 to 2014 in a newer version (NSIDES).

- **De novo prediction for drugs and side effects.** The question is: *Considering a novel DDI triplet (di,dj,rl), how good is a method at predicting adverse DDIs when one, two or three of the entities in the DDI triplet do not appear in the DDI dataset (that I use for training)?* As mentioned in Chapter 2, this is known as the "cold start" problem. The earlier work by Qian et al is the only one that addressed this problem, but their method was able to predict only the existence of an interaction, while DCSE-twin can predict the specific side effect.

- **"True" polypharmacy side effect prediction.** The question is: *for a given drug pair, how good is a method at predicting side effects that are not already caused by either of the 2 drugs alone?* This case is, arguably, the most significant in terms of assessing the risks of polypharmacy therapies.

The experiments presented in the sequel provide further proof that the distributed representations of the single drugs combine nonlinearly into the representation for the drug pair. They also quantify the improvement in the prediction of DDI obtained by including off-label single drug side effect data, and they highlight the role of the biases in modelling drug and side effect popularity.

5.1 Experimental Results

5.1.1 Datasets

The main sources of data I used for my work are the TWOSIDES and OFFSIDES datasets [54], which contain information about adverse events caused by the co-administration of two drugs, and off-label side effects caused by single drugs, respectively. TWOSIDES includes $1,318$ side effect types across $63,473$ drug pairs. There is also a total of 645 drugs that make up the different drug pairs. In total, there are $4,543,686$ associations between side effects and combinations of drugs. On average, each side effect is associated with $4,719$ drug pairs. Figure 5.1 shows the distribution of the number of drug pairs linked to each side effect. For a closer inspection of both ends of the plot, tables 5.1a and 5.1b illustrate the top and bottom 10 side effects, respectively, in the TWOSIDES dataset in terms of drug pairs that are associated with them.

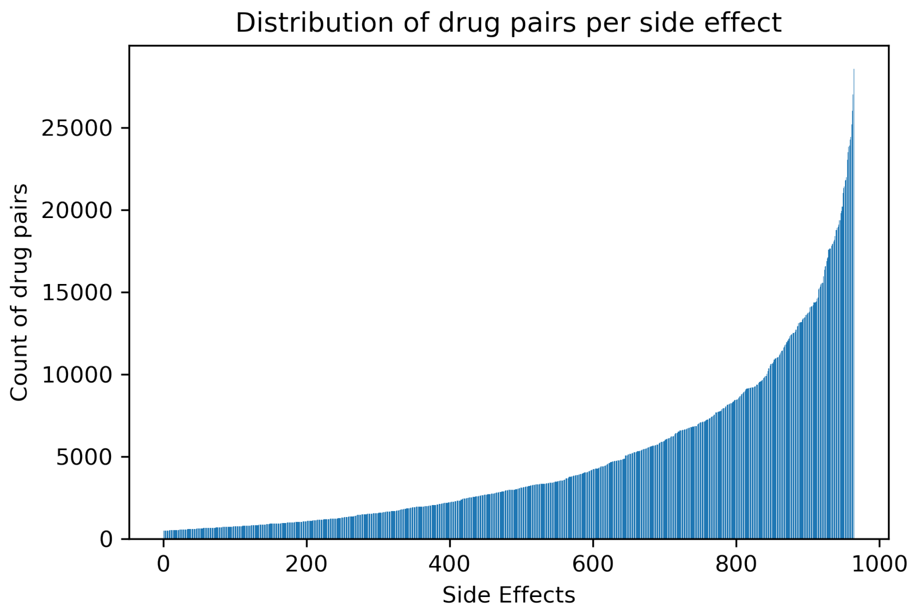


Figure 5.1: Distribution of the number of drug pairs associated with each side effect. On the x-axis we have the side effects arranged in ascending order, in terms of the number of drug pairs that cause them, and the y-axis represents the count of drug pairs that cause each side effect.

OFFSIDES contains information about $438,801$ associations between $10,097$ off-label side effects and $1,332$ drugs. The term “off-label” refers to any drug effect that is not already listed on the drug’s package insert. The average drug label lists 69 “on-label” adverse events. For OFFSIDES, there is an average of 329 high-confidence off-label adverse events for each drug.

Both TWOSIDES and OFFSIDES are the result of a data-driven approach that flagged significant drug-event associations from the AERS which collects reports

Table 5.1: Top and bottom 10 side effects in TWOSIDES. The first column represents the ranking in the top or bottom 10, the second column is the name of the side effect, and the last column is the number of drug pairs that are associated with the side effect.

(a) Top 10 side effects in TWOSIDES.

#	Side effect	Drug pairs
1	Arterial pressure decreased	28,568
2	Anaemia	27,006
3	Difficulty breathing	26,037
4	Nausea	25,190
5	Pneumonia	24,430
6	Fatigue	24,260
7	Pain	23,894
8	Diarrhoea	23,848
9	Asthenia	23,515
10	Emesis	23,043

(b) Bottom 10 side effects in TWOSIDES.

#	Side effect	Drug pairs
963	Sarcoma	502
962	Collagen disease	504
961	Neonatal respiratory distress syndrome	507
960	Impetigo	507
959	Coccydynia	508
958	Carbuncle	509
957	Atypical mycobacterial infection	510
956	Anaemia hypochromic	510
955	Multiple personality disorder	514
954	Normal pressure hydrocephalus	515

from doctors, patients and drug companies. In this approach, the authors were able to successfully remove synthetic associations from indications, co-prescriptions, and hidden covariates. The method used to build these datasets is called Statistical Correction of Uncharacterised Bias (SCRUB).

I also carried out a prospective evaluation using a newer version of TWOSIDES, called NSIDES [55]. TWOSIDES and OFFSIDES include adverse events reports up to and including 2009, whereas, NSIDES considers reports up to and including 2014. To the best of my knowledge, this is the first work that is actively making use of this newly released data. I used NSIDES to define a novel experimental setting to evaluate if my approach is able to prospectively predict newly discovered DDIs while training only with the older data. NSIDES contains information about $653,352$ new drug-drug interactions previously unknown in TWOSIDES. Note that only the interactions are new — i.e., the drug-drug-side effect triplets. The drugs and side effects involved in the triplets were already present in TWOSIDES and have interactions with other drugs and side effects.

5.1.2 Setting A - Split per side effect

The first experimental setup is the one that is adopted most commonly in the literature [1, 3, 56]. It was introduced by Zitnik et al. [1], and it is focused on predicting which pairs of drugs will cause a given side effect.

First, to make sure that there is a minimum number of information for each side effect, the TWOSIDES dataset is reduced so that only the side effects that occur for at least 500 drug pairs are kept. Then, for each side effect, I split the associated drug pairs into training, validation and test sets. Both the validation and test sets include 10% of the drug pairs and the training set is composed of the remaining 80%. The validation set is used to select the model parameters.

For the negative samples, I followed the exact same sampling procedure as described in the work by Zitnik et al. [1]. For each positive sample (d_i, d_j, r), I sampled a random negative sample (d_i, d_n, r) by randomly choosing drug d_n. This drug is selected randomly according to a sampling distribution P_r [26]. This sampling distribution P_r is an empirical adjustment of the "unigram distribution". The unmodified distribution is expressed by the following equation:

$$P\left(d_i\right) = \frac{f\left(d_i\right)}{\sum_{j=0}^{m}\left(f\left(d_j\right)\right)} \tag{5.1}$$

Where $f(d_i)$ is a function that returns the number of drugs with which drug i interacts with and causes side effect r. In network terms, $f(d_i)$ would mean the degree of node i. For instance, the probability of choosing the drug "Ibuprofen" as a negative sample, would be equal to the number of drugs with which "Ibuprofen" interacts with, divided by the total number of interactions for a given side effect. Intuitively, drugs that have many interactions with other drugs are more likely to be selected as negative samples. Nevertheless, the authors state in their paper that, empirically, the best adjustment to the equation above is to raise f to the power of 3/4, as follows:

$$P_r\left(d_i\right) = \frac{f\left(d_i\right)^{3/4}}{\sum_{j=0}^{m}\left(f\left(d_j\right)^{3/4}\right)} \tag{5.2}$$

Raising f to this power has the effect of increasing the probability for drugs with a small number of interactions and decreasing the probability for those with a high number of interactions. This behaviour is desirable, I don't want to pick high-interactive drugs all the time as negative samples since they are less valuable and less likely of being true negatives as the unusual or less-interactive drugs.

Since I sample one negative per positive sample, the final training set is balanced. For both validation and test sets, I kept the same balanced setting but the negative examples are sampled in a different way. For each positive sample, the drug d_n is selected randomly according to a uniform distribution.

The task at hand is then to predict drug combinations that are associated with each side effect type. I evaluated the performance of my approach against the methods detailed in Chapter 2. In addition to them, I also compared to a state-of-the-art machine learning approach, XgBoost. For this approach I built a feature vector for each drug based on PCA representation of the drug-target protein interaction matrix. I also combined the PCA representation of side effects of individual drugs from the OFFSIDES dataset. Drug pairs are represented by concatenating the corresponding drug feature vectors and used as input for XgBoost, that predicts the exact side effect of a pair of drugs. A different model is trained for each side effect type. Comparing against a state-of-the-art classifier is really important, especially since the feature vector is built from single drug side effect information, which is the same data I use for DCSE-left. Another approach used in the comparison is TransE [57]. This approach considers the problem of embedding entities and relationships of multi-relational data in low-dimensional vector spaces. It was used as a competitor in the TriVec paper, and it represents a good baseline for this problem. This method models relationships by interpreting them as translations operating on the low-dimensional embeddings of the entities. It is a state-of-the-art link prediction method on knowledge bases.

The parameter setting for the methods is determined using a validation set with a grid search over different possible values. For the methods that are not graph-based, the ideal parameters were selected based on the best performance on the validation set for each side effect type. The performance was calculated using the AUROC and the AUPRC. Higher values always indicate better performance. For those methods for which there is a different model per side effect type, the performance is averaged across the number of side effects.

From the results in Table 5.2, we see that DCSE-twin outperforms all other

Table 5.2: Performance comparison for **Setting A**. AUROC and AUPRC are shown for all methods.

Method	AUROC	AUPRC
XgBoost*	0.801	0.811
DeepWalk*	0.715	0.709
DeepDDI*	0.623	0.610
Decagon†	0.872	0.832
TransE	0.900	0.863
TriVec	0.905	0.869
DCSE-twin	**0.927**	**0.882**

$\star$ Performance is averaged across 963 side effects.
$\dagger$ As reported in [1].

methods. The difference with DeepDDI possibly indicates a potential limitation in relying only on chemical-based features to represent drugs for the prediction of polypharmacy side effects. It also hints that just using deep neural networks is not enough to achieve competitive performance in DDI prediction. Both DeepWalk and XgBoost show an improvement over DeepDDI. These methods first learn the representation for the drugs and then use these learned features to predict the associations. The fact that the learning in these two stages is kept separate, highlights that there is an interdependence between the different side effects and the combination of drugs that are not being addressed in these models. These findings show that end-to-end learning can greatly benefit performance. When comparing against the other end-to-end methods, namely, Decagon, TransE and TriVec, DCSE-twin still shows a clear advantage in performance gains. It demonstrates that the distributed representations learned by my approach are highly descriptive and are able to outperform other methods that rely on graph convolutions and different biological and chemical features. Learning a representation for the drugs and side effects that effectively captures how they are interconnected, provides a lot of flexibility and simplicity to my approach. It also shows that the information available about the single drugs and polypharmacy side effects in OFFSIDES and TWOSIDES, respectively, is rich enough to learn a system that is able to outperform other highly complex methods that rely on a lot of external information.

5.1.3 Setting B - *De novo* prediction for drug pairs

While Setting A focuses primarily on the drug pairs that are associated with specific side effects, for this new setting I focused, instead, on all the side effects caused by the drug pairs. Specifically, the aim is to predict which side effects, out of all the possible ones, are caused by the interaction between the two drugs. To achieve this, the testing split is done per drug pairs where I held out all the side effects associated with the combination, for testing.

This experimental setup is meant to frame the problem as a *de novo* prediction of side effects for "new" or unseen drug pairs. These experiments can provide a better assessment of the real-life utility of the different methods. To my knowledge, I am the first to introduce this setting and this represents another important contribution of my work. I believe this experimental setup has a real value as it represents the problem in a different and, possibly, a more realistic way.

In order to formally understand this setting, let's define a new matrix $S \in \mathbf{R}^{C \times n}$. Where C is the number of known drug combinations in TWOSIDES with at least one associated adverse side effect, and n is the number of side effects. For these experiments, I used the entire TWOSIDES dataset (I removed the constraint I had in Setting A, where I only keep side effects with at least 500 drug pairs associated with them. In total, the number of side effects (n) is $1,318$, and the number of drug pairs (C) is $63,473$).

To split the data into training, validation and testing, I randomly selected 10% of the rows in S and use this as the testing set. These rows represent all the side effects associated with the drug pairs. I did the same for the validation set, by selecting another 10%. I used the remaining 80% of the rows to train the model. I took all the unknown entries in S as the set of negative samples. This makes the problem highly unbalanced in terms of the negatives to positives ratio, having a density[1] of only 7%. This selection of negative samples made the most sense because my aim is to assign a probability score for all possible side effects for a given drug pair.

I chose DeepDDI as the competitor for this setting because it's the method that can be adjusted better to this experimental setup. This method also has some characteristics that make it an ideal baseline to compare against my work. It is highly nonlinear as it employs deep neural networks like my own approach. And it makes use of features that depend on the chemical structures of the drugs as input. This last aspect can also illustrate how a method that relies on engineered features compares to my approach. To make DeepDDI work for this setting, the only change I made was to the output layer. Instead of having one output, I changed it to have n outputs, where n is the number of side effects I am predicting for. The model then becomes a multi-label classifier that gives a probability score for each side effect, from the input features of the drug pair.

For setting B, I first evaluate the performance of DCSE-twin and DeepDDI by measuring the AUROC and AUPR in the test set. From Table 5.3 we can observe that, under these metrics, my method shows a clear improvement over the DeepDDI method. It is important to note that DCSE-twin still gets a clear edge over DeepDDI,

[1]The density of a matrix is defined as the ratio of nonzero entries to the total number of elements. For this problem, it means the total number of 1_s in S, divided by the total number of elements, given by $C \times n$.

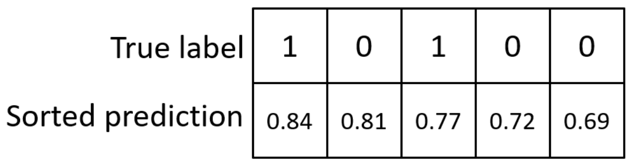
TABLE 5.3: Performance comparison for **Setting B**. AUROC and AUPRC are shown for both methods.

Method	AUROC	AUPRC
DeepDDI	0.767	0.220
DCSE-twin	**0.851**	**0.360**

in spite of not depending on any type of biological features.

I also used information retrieval metrics, like the Precision and Normalised Discounted Cumulative Gain in the top-10 (Prec@10 and NDCG@10, respectively) to measure the performance. These metrics were computed for each drug pair or row in the testing set. The precision at the top-k, measures the proportion of items in the top-k set that are relevant, or true ones in the test set. Formally, let's define Prec@k as follows:

$$\text{Prec@}k = \frac{\mid \{\text{ relevant items }\} \cap \{k \text{ retrieved items }\} \mid}{\mid \{k \text{ retrieved items }\} \mid} \tag{5.3}$$



True label	1	0	1	0	0
Sorted prediction	0.84	0.81	0.77	0.72	0.69

FIGURE 5.2: Example of the top-5 predictions for a row in the testing set for Setting B. The predicted scores are sorted in descending order, as represented in the bottom row. The top row represents the true labels for the top-5 side effects.

To illustrate how to calculate the precision, let's look at the simple example shown in Figure 5.2. Following this example, let's calculate Prec@5 by applying Equation 5.3. The relevant items in the top-5 retrieved items are 2. This means that the Prec@5 : $\frac{2}{5} = 0.4$.

DCG provides a performance of ranking by penalising relevant items ranked lower than non relevant items. Intuitively, it takes into account the ranking of the true ones retrieved in the top-k items. DCG is defined as:

$$\text{DCG@}k = \sum_{i=1}^{k} \frac{rel_i}{\log_2(i+1)} = rel_1 + \sum_{i=2}^{k} \frac{rel_i}{\log_2(i+1)} \tag{5.4}$$

Where rel_i represents the relevance of item i. In a binary setting, it is either 1 or 0. Note that we can start the summation at position 2, since the denominator for $i = 1$ is equal to 1. DCG penalises highly relevant items that appear lower in the search by reducing the graded relevance value logarithimically proportional to the position of the result. Following the example in Figure 5.2, the DCG@5 : $1+0+\frac{1}{\log_2(3+1)}+0+0 = 1.5$. The NDCG@$k$, is the normalised version of the DCG, it divides the DCG by the Ideal DCG (IDCG). IDCG is defined as the DCG of the ideal ordering. Again, if we follow the same example, the IDCG@5 : $1 + \frac{1}{\log_2(2+1)} + 0 + 0 + 0 = 1.63$. Finally, the NDCG@$k$ is defined as:

$$\text{NDCG@}k = \frac{\text{DCG@}k}{\text{IDCG@}k} \tag{5.5}$$

In our example, NDCG@k : $\frac{1.5}{1.63} = 0.92$. Note that the ratios will always be in the range of $[0, 1]$ with 1 being a perfect score, meaning that the DCG is the same as the IDCG. Therefore, the NDCG values can be averaged for all drug pairs or rows in the testing set, to obtain a measure of the average performance of how the different methods rank the results.

Table 5.4 shows the performance of DCSE-twin compared to DeepDDI under the information retrieval metrics. Again, my approach shows a clear advantage over DeepDDI. We can observe from the Prec@10 that for DCSE-twin, roughly half of the top-10 predictions are true ones. And out of these, from the NDCG@10, we

TABLE 5.4: Performance comparison for **Setting B** using information retrieval metrics. Precision and Normalised Discounted Cumulative Gain at the top-10 (Prec@10 and NDCG@10) are shown for both methods. Results are averaged throughout all drug pairs in the testing set.

Method	Prec@10	NDCG@10
DeepDDI	0.389	0.667
DCSE-twin	**0.462**	**0.725**

know that they are ranked, on average, higher than the negative examples. These results are very encouraging, as they show a very solid performance for combinations of drugs without any previous information on interactions between them. Having computational methods that can accurately predict side effects for new drug pairs that are being tested for approval, can reduce the time and cost for the drug development pipeline [58]. In particular, the remaining predictions in the top-10, which are not true ones, are prime candidates for further analysis and experiments.

5.1.4 Prospective Evaluation

I evaluated the performance of DCSE-twin through a simulated prospective approach. Prospective evaluation is a method that preserves the chronological order in which the information became available historically. It is crucial to preserve the chronological order because a method that predicts side effects, needs to be able to predict an unknown association based only on the information available before the association becomes known. Using different approaches for testing like cross-validation or Settings A & B, can potentially break the chronological order by including in the training set drug-drug-side effect associations that historically became known only after related associations in the testing set became known [59]. This can provide an unfair and unrealistic advantage to the prediction methods.

To the best of my knowledge, this is the first work that introduces a prospective evaluation for DDI prediction.

For the evaluation, I trained the models on DDI data up to 2009 (TWOSIDES) and used them to predict the new associations reported from 2010 to 2014 (NSIDES). This setup can be applied to both previous settings, A and B. I subset NSIDES to account only for the same drug pairs and side effects that occur in TWOSIDES. After filtering out the extra drugs and side effects from NSIDES, there is a total of $653,352$ new drug-drug-side effect triplets distributed among 764 different side effects. It is important to mention that the sampling of the negative triplets for the training set was done independently from the prospective testing set. Any prospective positive DDI from the testing set can be sampled as a negative for the training set since at the time there was no information about these associations. In this way, we are preventing data leakage between the training and the prospective testing set.

Table 5.5: Performance comparison of the prospective testing set for Setting A.

Method	AUROC	AUPRC
DeepDDI	0.607	0.611
XgBoost	0.798	0.804
DCSE-twin	**0.841**	**0.885**

For Setting B, let's remember that the testing is done per drug pair, not by side effect, so, the setting is different than for Setting A. I kept from NSIDES the same drugs and side effects that are present in TWOSIDES. This allowed me to identify new drug pairs in NSIDES that previously had no interactions in TWOSIDES. The total number of new drug pairs in our prospective testing set for Setting B was $5,430$, with an average of 14 side effects per drug pair. I also made sure that there are at least 5 side effects associated with each drug pair.

Table 5.6: Performance comparison of the prospective testing set for Setting B.

Method	AUROC	AUPRC	Prec@10	NDCG@10
DeepDDI	0.844	0.091	0.129	0.401
DCSE-twin	**0.871**	**0.145**	**0.175**	**0.412**

DCSE-twin achieved the highest AUROC and AUPRC for the prospective testing in both Settings A and B, as can be observed in Tables 5.5 and 5.6. These findings suggest that the distributed representations, learned by DCSE-twin, are descriptive enough to generalise for future reported drug pair-side effects associations at a higher level than other methods that rely on biological and chemical features. The goal of this evaluation was to demonstrate the real-life utility of my proposed method by testing in a realistic scenario where the samples in training and testing follow a chronological order. This approach can complement existing hypotheses that support the work of drug safety and drug development professionals. The practitioners can follow up on the highest-scoring predictions, for a particular new drug pair or side effect, with a thorough clinical investigation to focus their limited resources on hypotheses that have a higher probability of being true.

5.1.5 *De novo* prediction for drugs and side effects

A key aspect of my work is the addition of DCSE-left and the data from the OFFSIDES dataset. This opens up new possibilities for different research questions that frame the problem under novel scenarios with important biological implications. The idea for this evaluation was to make *de novo* predictions of adverse DDIs for unseen drugs and/or side effects, by utilising the learned distributed representations available only to DCSE-left. Specifically, there are 697 and $9,134$ extra drugs and side effects, respectively, that are present only in OFFSIDES and not in TWOSIDES.

Thus, my approach is solving the *cold start* problem for adverse DDI prediction for all of these extra drugs and side effects.

Table 5.7: Different scenarios for *de novo* prediction of DDIs for new drugs and side effects. All of the drugs and side effects come from the OFFSIDE dataset. The seen/unseen is to specify if a drug or side effect was seen/unseen during training by DCSE-right, i.e. are present in TWOSIDES.

Scenario	**Drug 1**	**Drug 2**	**Side Effect**
1	Seen	Seen	Unseen
2	Seen	Unseen	Seen
	Unseen	Seen	Seen
3	Seen	Unseen	Unseen
	Unseen	Seen	Unseen
4	Unseen	Unseen	Seen
5	Unseen	Unseen	Unseen

Qian et al. [43], considered in their work the scenario of predicting interactions of drugs without any prior information about their interaction profile during training. They showed the importance of these predictions and highlighted the inability of most current methods of predicting DDIs for newly developed drugs. In my work, I extended this notion by making *de novo* predictions for unseen drugs during training, and also providing the specific adverse side effect caused by the combination. Importantly, the side effects can also be unseen during the training phase. To the best of my knowledge, DCSE-twin is the first method that can predict for unseen side effects, as the current methods in the literature can't account for this possibility.

First, let's understand why I am able to make these *de novo* predictions for drugs and side effects that are not seen by DCSE-right. In my approach, DCSE-right takes as input two drugs and a side effect and outputs a score that can be interpreted as the probability of the side effect being the consequence of the interaction between the two drugs. However, after the training phase of the model is finished, I only need to input the low-dimensional vectors of drugs and side effects to obtain a prediction. Even if the drugs or side effects were not seen by DCSE-right, and I can't input the one-hot encoding, the combination layers will still be able to output a distributed representation for the drugs, using Eq. 4.2. After getting the vector that represents the new drug pair, the prediction can easily be obtained by applying Eq. 4.14.

To evaluate the performance of DCSE-twin when predicting these new DDIs, I used the data from NSIDES to verify which of these associations became true after OFFSIDES and TWOSIDES were released. Since we have two drugs and a side effect as input to my model, there are different possibilities and scenarios for this *de novo* predictions. Table 5.7 shows a breakdown of the different scenarios. The first scenario occurs when both drugs are seen during training, but a new side effect is introduced. This scenario can prove to be useful for monitoring drugs in their postmarketing stage to detect new risks when they are exposed to a larger patient population. It can also serve as a decision-making tool for physicians when prescribing combinations of existing drugs. Scenario 1 can also reduce time for the approval of a new repurposing strategy of combinations of known drugs. For this setting, there are currently no existing methods that can make predictions for side effects that lack any known associations with drug combinations. The next scenario is when one of the two drugs is unseen, and the other drug and the side effect are both seen. In this case, we start introducing some degree of difficulty to the prediction problem. The following scenario also has one of the two drugs unseen, but also the side effect is now unseen during training. For postmarketing surveillance of drugs, these last two scenarios can offer a good assessment to tell if a method is useful or not to detect DDIs for recently approved drugs when combined with well established and known drugs. This surveillance refers to the monitoring of drugs after they reach the market [60]. In scenario 4, both drugs are unseen during training, and only the side effect has some other drugs pairs associated to it. This scenario is particularly interesting for a new drug combination in development. Identifying DDIs at the preclinical stage helps drug safety professionals allocate resources and take appropriate regulatory action [61]. This type of experiments can help guide experts in designing clinical trials for the verification of these side effects. The last scenario is where both drugs and the side effect are unseen during training. This means that neither of the drugs or side effects have any form of interaction profile known at the time TWOSIDES and OFFSIDES were released. This is the hardest scenario to predict for.

Several existing methods suffer from the *cold start* problem and can't provide a prediction for new drugs or side effects. Most of these methods are graph-based [1, 3] where there is no information to capture from the knowledge graph for disconnected nodes (new drugs) or link types (new side effects) without any nodes linked together. However, feature-based methods such as DeepDDI and XgBoost, can make *de novo* predictions as long as the features are available for the new drugs. A limitation of these two methods is that the side effect is not an input to the model. This means that they can't make predictions for the scenarios in which the side effect hasn't been seen during training (Scenarios 1, 3 and 5). For this reason, I only compared the performance of my method against DeepDDI and XgBoost in Scenarios 2 and 4. Table 5.8 shows the performance for each of the different scenarios and competing methods.

In Scenarios 1, 3 and 5, where the side effect is completely new, DCSE-twin achieved good results in terms of AUROC and AUPRC. The performance was, as expected, highest when both drugs were seen during training and lowest where

TABLE 5.8: Performance for the different *de novo* prediction scenarios. AUROC and AUPRC are shown for each of the scenarios and competing methods. The best performance is shown in bold.

Scenario	**Drug 1**	**Drug 2**	**Side Effect**	**Method**	**AUROC**	**AUPRC**
1	Seen	Seen	Unseen	**DCSE-twin**	**0.815**	**0.778**
2	Seen	Unseen	Seen	DeepDDI	0.784	0.743
				XgBoost	0.801	0.774
	Unseen	Seen	Seen	**DCSE-twin**	**0.845**	**0.831**
3	Seen	Unseen	Unseen	**DCSE-twin**	**0.773**	**0.756**
	Unseen	Seen	Unseen			
4	Unseen	Unseen	Seen	DeepDDI	0.817	0.799
				XgBoost	0.808	0.771
				DCSE-twin	**0.833**	**0.815**
5	Unseen	Unseen	Unseen	**DCSE-twin**	**0.731**	**0.697**

neither drug nor the side effect was seen. For the rest of the scenarios in which I can compare against two different baselines, DCSE-twin still obtained the highest performance. These results indicate that the distributed representations learned only by DCSE-left and used for predictions in the right, contain enough information to make robust *de novo* predictions for unseen drugs and side effects.

Many of the side effects for drugs, and especially for drug combinations, are recognised after the drug was approved and already in the market [24]. SD methods are deployed to detect safety concerns for marketed drugs. These concerns may be new in nature, such as new adverse side effects, not recognised during clinical trials in the premarketing phases. Quantitative SD methods typically adopt machine learning approaches to provide new hypotheses for experts to investigate further. The experimental results for these different scenarios highlight the value for methods, such as my work, to be deployed as SD methods in postmarketing surveillance for drugs and side effects. The aim of these methods is to discover potential safety concerns, either completely novel (Scenarios 1, 3 and 5) or new patterns to previously known adverse drug reactions (Scenarios 2 and 4). They can also be used to validate or confirm signals that have been identified from different avenues, like case-by-case assessments of individual case safety reports.

5.1.6 Predicting *true* polypharmacy side effects

Many of the side effects that are associated with a drug pair are already linked to either one of the single drugs or to both of them. However, there are side effects that are only due to the interaction between the pair of drugs. These side effects are of particular importance since they don't occur when only one of the single drugs is being administered. For example, the example from Chapter 1 in Fig. 1.1, illustrates this case: Aspirin and Warfarin can have the side effect of *gastrointestinal haemorrhage* when the two drugs are prescribed together, but not when they are taken individually.

In TWOSIDES there are 868,221 significant associations that cannot be clearly attributed to either drug alone. Evaluating the performance when predicting these associations will tell us more accurately how good a method is at predicting *true* polypharmacy side effects. This experiment is being overlooked in the current literature on DDI prediction.

To find these *true* polypharmacy side effects, I compared TWOSIDES with OFFSIDES and analysed which side effects for a given drug pair, occur only in TWOSIDES and not in OFFSIDES for any of the single drugs. On average, 67% of the side effects for a drug pair only occur due to the interaction between the drugs. Figure 5.3 shows the full distribution of the proportion of the side effects per drug pair that are considered *true*. This means that this proportion of side effects are present in TWOSIDES, but not in OFFSIDES for any of the drugs on their own.

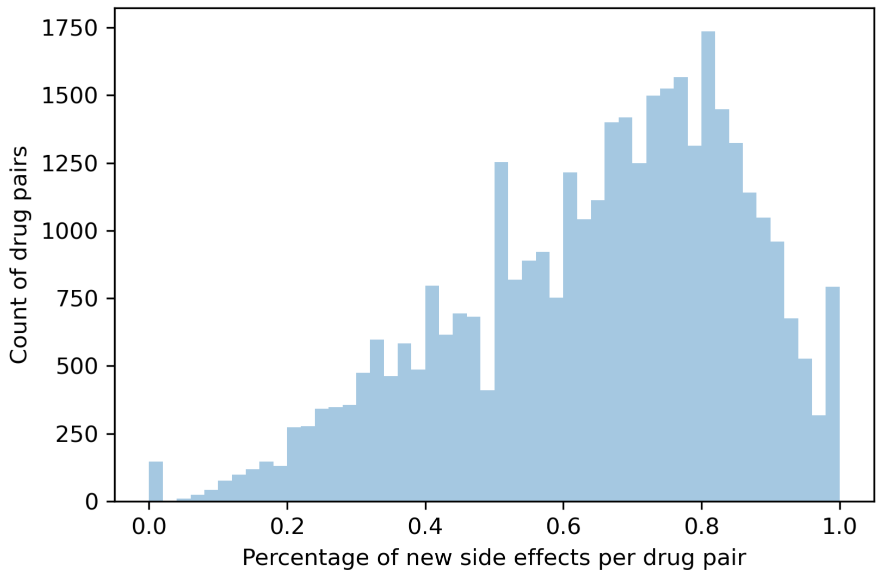


FIGURE 5.3: Histogram of the proportion of *true* polypharmacy side effects per drug pair. The height of each bar, in the y-axis, represents the count of drug pairs that have a specific proportion of *true* side effects, as shown in the x-axis.

For this experiment, I used the same training set as in Setting A. To build the testing set I selected only the *true* polypharmacy side effects as positive samples and keep all the existing negative samples. Figure 5.4 shows a graphic representation of this process. An important observation is that in the dataset made available by Zitnik et al [1], in the Decagon paper[2], they considered all possible side effects, even the ones that can already be explained by either of the individual drugs. This is in contrast to what they mention in the paper, where they argued that the predictions by Decagon are made only for true polypharmacy side effects

[2]http://snap.stanford.edu/decagon.

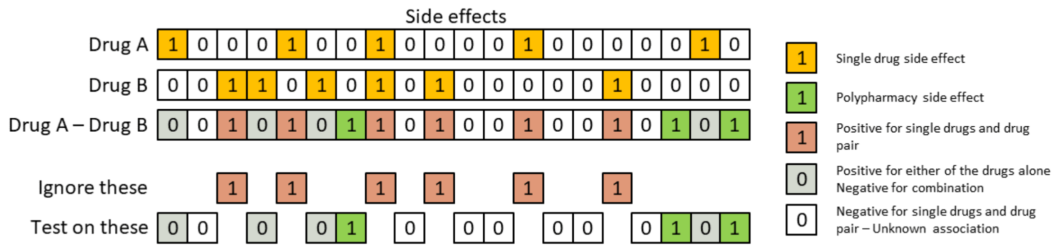

Side effects
Drug A 1 0 0 0 1 0 0 1 0 0 0 0 1 0 0 0 0 0 1 0
Drug B 0 0 1 1 0 1 0 1 0 1 0 0 0 0 0 1 0 0 0 0
Drug A – Drug B 0 0 1 0 1 0 1 1 0 1 0 0 1 0 0 1 0 1 0 1
Ignore these 1 1 1 1 1 1
Test on these 0 0 0 0 1 0 0 0 0 0 0 1 0 1
Single drug side effect
Polypharmacy side effect
Positive for single drugs and drug pair
Positive for either of the drugs alone Negative for combination
Negative for single drugs and drug pair – Unknown association

Figure 5.4: Testing on true polypharmacy side effects. Drugs A & B cause some side effects on their own that are marked in yellow. Some of these side effects still occur for the drug pair, these are marked in red. However, the combination does exhibit some new side effects that can only be attributed to the pair of drugs and not by any of them individually. These side effects are marked in green. Lastly, some of the side effects caused by either of the drugs on their own, no longer happen for the drug pair, these cases are marked in grey. These 0_s hold more value than the white 0_s as they represent a negative association between the pair and the side effect. Whereas, the white 0_s are unknown in the sense that they can either mean a negative association but there's not enough information to verify this. The idea is to test only on the side effects that are caused by the drug pair and not by any of the single drugs on their own. This means that I will ignore the red 1_s and just test on the green 1_s. Both the grey and white 0_s can still be used for negative samples.

From Table 5.9 we can observe that my method outperforms the other approaches when predicting only *true* DDIs. I believe the importance of these *true* DDIs cannot be overstated. They hold a great biological and medical significance. It can help in the design of clinical trials and also in the prescription of new combination therapies for complex diseases. Note that the red ones and grey zeroes in Figure 5.4 are also interesting cases but are not being explored in this work. The red ones represent side effects that happen for one or both drugs and are being exacerbated by the interaction. And the grey zeroes can either be side effects that stopped occurring for the combination, or just that the frequency with which the side effect occurs for either of the two drugs on their own is not being exacerbated by the drug interaction.

TABLE 5.9: Performance comparison for the different methods when testing only on *new* DDIs. The side effects caused by the drug pairs in this set are the result of the interaction between the two drugs and are not caused by any of them on their own.

Approach	AUROC	AUPRC
DeepDDI	0.617	0.594
DeepWalk	0.693	0.765
XgBoost	0.788	0.804
My method	**0.910**	**0.823**

5.2 The distributed representations for the drug pairs are learned nonlinearly

One of the fundamental pieces of my work is the MLP which learns how to combine the distributed representations of the individual drugs to learn the representation for the drug pair. The findings from Section 3.3 made it clear how a simple linear model is unable to fully capture the interaction between the drugs. This information allowed us to gain a better understanding of the biological implications of the problem, namely, that nonlinearity is necessary to model how two drugs combine. However, in that early method, the experimental setting was different as I was comparing how a linear model would be able to learn a representation for drug pairs starting from the already learned and pre-trained representations for the individual drugs. For this experiment, I wanted to show how DCSE-twin performs when I removed the MLP from DCSE-right when compared to my full method. The idea is to not only prove that DCSE-twin works better with the MLP, but to show that to model the interactions between two drugs, a simple linear model is not sufficient.

For this experiment, I removed the nonlinear combination layers from DCSE-right and performed an ablation study [62]. This type of experiments are important to understand if the main components in a model are actually related to the initial hypothesis. Essentially, the architecture is reduced to a single linear layer, above the individual drug representations, that will learn coefficients to learn the representation for the combination of the two drugs, see Figure 5.5. The rest of the model remains exactly the same, including DCSE-left, to keep the experiment fair.

Table 5.10 shows how wide the gap is between my full approach versus the one without the nonlinear combination layers. This result confirms my initial hypothesis, introduced in Section 3.3, that a nonlinear approach is required to model the complex interaction between the drugs and learn the interplay between them.

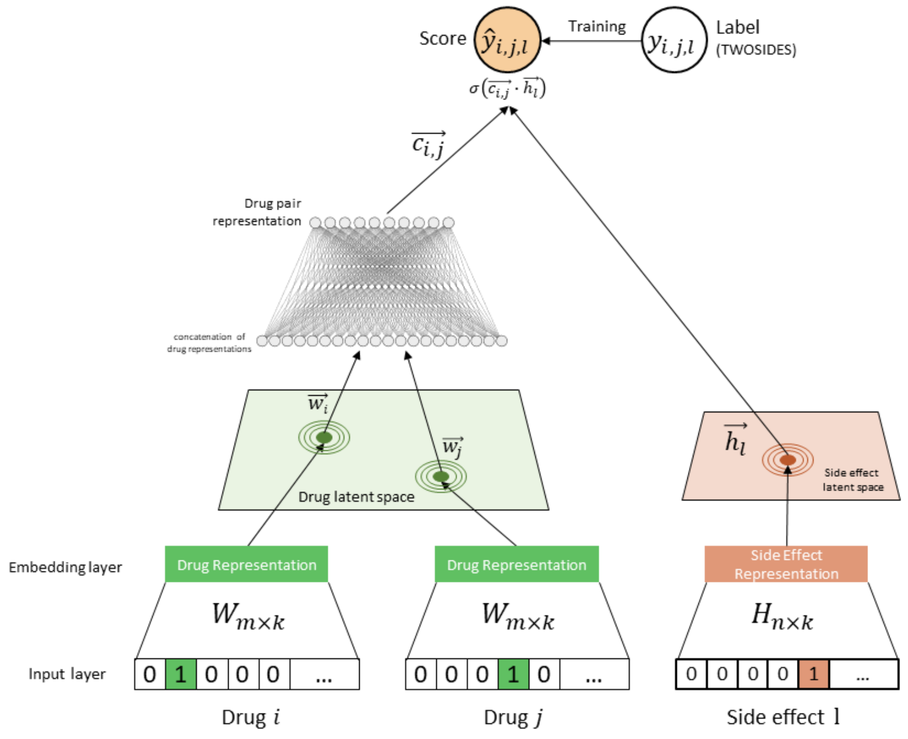


FIGURE 5.5: Right-side model architecture without the combination MLP layers.

Table 5.10: Ablation study - Performance of the approach without the combination layers (MLP) compared against the full model.

Approach	AUROC	AUPRC
Model with no MLP	0.784	0.775
Full approach	**0.912**	**0.873**

5.3 Importance of the off-label single drug side effect data

DCSE-twin is enriched greatly by the addition of DCSE-left and the corresponding data from OFFSIDES. Incorporating this model into my approach allows to utilise more data to learn better and more descriptive distributed representations. There are two main reasons behind this: it opens up new ways of framing the problem to answer different interesting biological questions; and to obtain better predictive performance. I have already discussed the first reason in section 5.1.5. Now, to understand if DCSE-left is impacting the performance of my model I performed an ablation study where I tested my method without training with the data from OFFSIDES. For this experiment, I used the same preparation as in Setting A for the testing set.

First, I trained just DCSE-right (as in Fig. 4.2) for a fixed number of epochs. This number was set to 20 for the purpose of this experiment. I used the trained model to predict the DDIs in the testing set and then calculated the performance. As seen in Table 5.11, DCSE-right on its own achieves comparable performance with the full approach.

Although the overall performance was not increased by a wide margin when I use DCSE-left, the benefits of using this model are not only limited to performance gains. By learning the distributed representations for a large number of drugs and side effects without any information about their interactions, I am able to provide *de novo* predictions of DDIs, using these representations. This is due to the fact that by learning them together with DCSE-right, the representations lie in the same latent space and can be used in any of the two models to obtain a prediction.

Table 5.11: Ablation study - Performance of DCSE-right alone compared against the full model.

Approach	AUROC	AUPRC
DCSE-right only	0.892	0.858
DCSE-twin	**0.912**	**0.873**

5.4 Are the biases, u and v, modelling drug and side effect popularity?

To understand exactly what the biases are learning, I explored the possibility that they might be modelling drug and side effect popularity. Intuitively, drug popularity in this context can be quantified by the number of side effects that are known to be associated with a given drug. Conversely, side effect popularity is quantified by the number of drugs that are known to the associated with a given side effect. The more information there is about a particular drug or side effect, we can say the more popular they are. Formally, I define the popularity d_i^{pop} for a drug d_i as the sum of the row i in X, and the popularity r_l^{pop} for a side effect r_l as the sum of the column l in X:

$$d_i^{pop} = \sum_j^{n+q} X_{i,j}, \quad r_l^{pop} = \sum_j^{m+o} X_{j,l}. \tag{5.6}$$

The hypothesis is that if the biases are modelling popularity, then u and v should be negatively correlated to the vectors of drugs and side effects popularity, $\vec{d}^{pop}$ and $\vec{r}^{pop}$, respectively. The first assumption is that the model should, generally, assign a higher score for drugs and side effects that are more popular. If that is the case, considering the nature of the non-normalised scores, then for higher values of u and v, the smaller the prediction score will be (see Eq. 4.7). Inversely, for lower values of u and v, we should obtain higher predicted scores.

The Pearson correlation obtained for $\vec{d}^{pop}$ and u is -0.831 (Fischer's two-sided exact test, $p < 2.68 \times 10^{-146}$), and for $\vec{r}^{pop}$ and v is -0.844 ($p < 1.52 \times 10^{-265}$). These values confirmed our hypothesis that the biases for the drugs and side effects are modelling popularity. The hope is that if the biases are already modelling this aspect from the data, then the factor modelling for the distributed representations are able to learn from more biologically meaningful interactions from the data and improve their descriptive power.

5.5 Reproducibility Analysis

In this section, I analysed if the distributed representations are reproducible across independent runs of my approach. This is an important evaluation to ensure that the biological interpretability of the representations is reproducible.

As a first evaluation, I followed the procedure adopted by Alexandrov et al. to study cancer mutational signatures [63, 64]. The reproducibility procedure is detailed as follows:

- *(Step 1)* Run the approach for $1,000$ times using all the available data from TWOSIDES and OFFSIDES with the length of the distributed representations

$k = 20$.

- *(Step 2)* Select the best 100 solutions that minimise the loss function and aggregate them into the matrices $\mathcal{W}$ of $m \times L$, and $\mathcal{H}$ of $n \times L$, where $L = 20x100 = 2,000$ latent features, $m = 635$ drugs and $n = 963$ side effects.

- *(Step 3)* Apply a partition clustering algorithm on the columns of $\mathcal{W}$ and $\mathcal{H}$, using the cosine distance as the metric[3]. Then, I run the k-means++ algorithm [65] with $k = 20$. The reproducibility of the distributed representations for the drugs and side effects is then calculated by the tightness and separation of the clusters obtained. I use the cosine-similarity based average silhouette width [66] of each cluster as a measure of reproducibility of each component of the representation. This width is a measure of how similar that component is to components in its own cluster, when compared to components in other clusters. The silhouette width s_i for the ith component of the latent representation is defined as:

$$s_i = \frac{b_i - a_i}{\max(a_i, b_i)} \tag{5.7}$$

  where a_i is the average distance from the ith point (a component of the representation) to the other points in the same cluster as i, and b_i is the minimum average distance from the ith point to points in different clusters, minimised over clusters. The silhouette width values range from -1 to 1. A value close to $+1$ indicates that a component is very similar to other components in its cluster but very dissimilar to neighbouring clusters.

[3]The cosine distance for two vectors w_1 and w_2 is defined as $d(w_1, w_2) = 1 - \frac{w_1 w_2^I}{\sqrt{(w_1 w_1^T)(w_2 w_2^I)}}$. The cosine similarity is $1 - d(w_1, w_2)$.

My results for the reproducibility analysis are shown in Figure 5.6. The silhouette plots shown in the figure is useful to study the separation distance between the clusters. They provide a way to assess the cluster quality visually. Ideally, the average silhouette score of all the clusters should be higher than 0.5 and as close to 1 as possible. Unfortunately, none of the components of the distributed representations for $k = 20$, for the drugs, have an average reproducibility score above 0.5. This analysis can also help guide the selection for the length of the distributed representations. To analyse this, Figure 5.7 shows the silhouette plot for the reproducibility analysis when the length of the representations is $k = 10$. As can be observed, the average score is increased from 0.08 to 0.26. It is still not the desirable score, however, two components of the representations do obtain a score higher than 0.5.

These results indicate that for the moment, the distributed representations are not fully reproducible across independent runs. This is mostly due to the fact that the values for the distributed representations are still changing too much after every epoch. To understand this behaviour I calculate the relative and average change in the weights of W and H through the epochs. Calculating the relative change in values is a standard stopping criterion during the learning of NMF methods [47]. If we define W_0 as the values of W from the previous iteration, and W as the current state of the latent representations, then the relative change is calculated as follows:

$$\frac{\max\left(W - W_0\right)}{\max\left(W_0\right)}. \tag{5.8}$$

The average change in values is the mean of the matrix: $W - W_0$. Figure 5.8 shows the analysis of the change in values for the traditional one-to-one learning pace of DCSE-twin. As can be observed, for both the relative and average change,

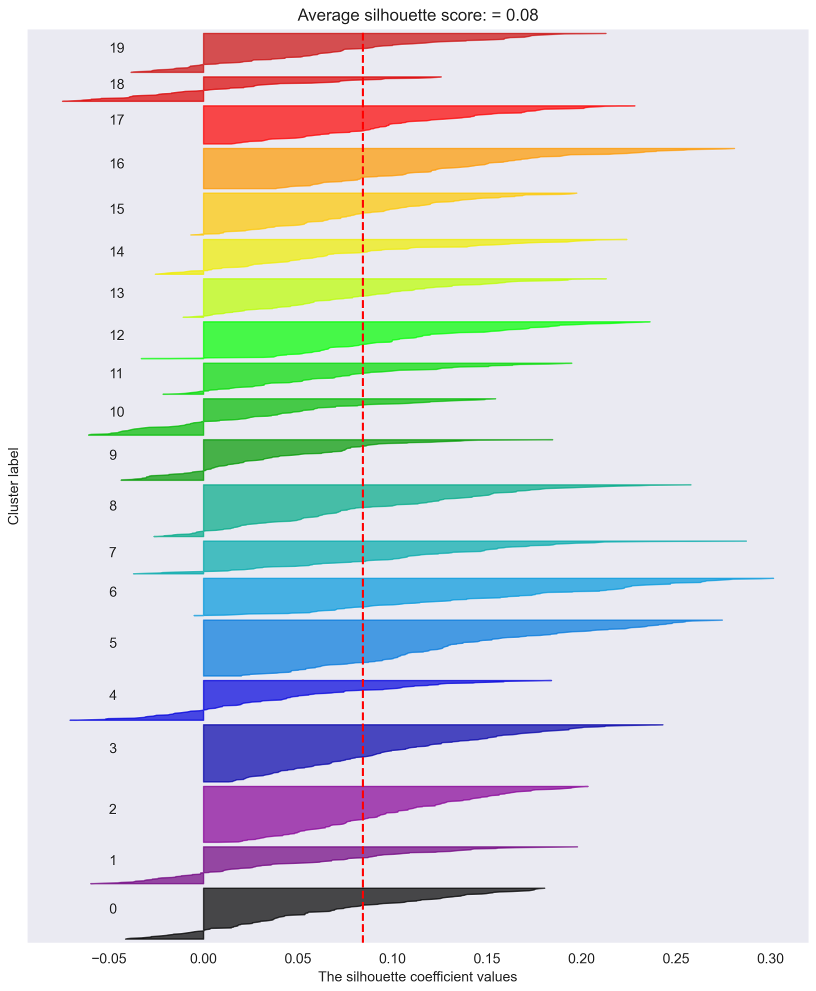


Figure 5.6: Silhouette analysis for the KMeans++ clustering of the drugs' distributed representations, for $k = 20$. In the x-axis we observe the clusters obtained by the algorithm and in the y-axis the range of silhouette coefficient values.

the values go down. Nevertheless, they get stabilised and stop decreasing in value. Ideally, the relative change shouldn't be higher than 1×10^{-3}, but the values are still not that close to this value. From the average change in W and H (Fig. 5.8 bottom), we can observe that the values go down and then start going up again. The struggle in the learning between the left-side and right-side models is more apparent

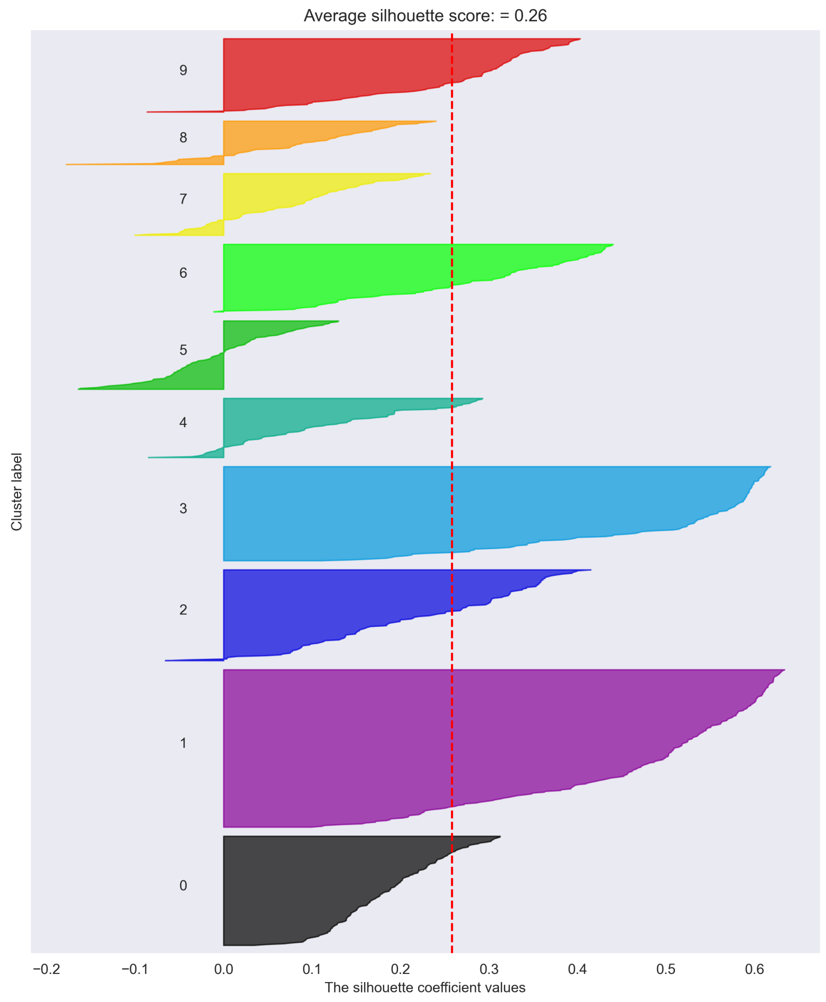


FIGURE 5.7: Silhouette analysis for the KMeans++ clustering of the drugs' distributed representations, for $k = 10$. In the x-axis we observe the clusters obtained by the algorithm and in the y-axis the range of silhouette coefficient values.

when we consider the overall change in values.

However, there are multiple things that I can adjust in DCSE-twin to make the representations more stable, and thus, reproducible. As discussed in Section A.2, freezing the weights of the representations and having only DCSE-left update them, greatly reduces the struggle between the learning of the two models. It also has the added effect of stabilising the values of the representations. In Figure 5.9 we can observe the analysis of the change in values when I *freeze* the weights of W and H during the learning of DCSE-right. The relative change gets to a lower value for both W and H. And the average change is also more stable. For future work, I need to analyse different combinations of parameters like the number of times I only learn the weights of W and H in DCSE-left before starting the learning of the right-side. The number of *frozen* epochs and the size of the distributed representations may also play a part in decreasing their overall change in values between epochs.

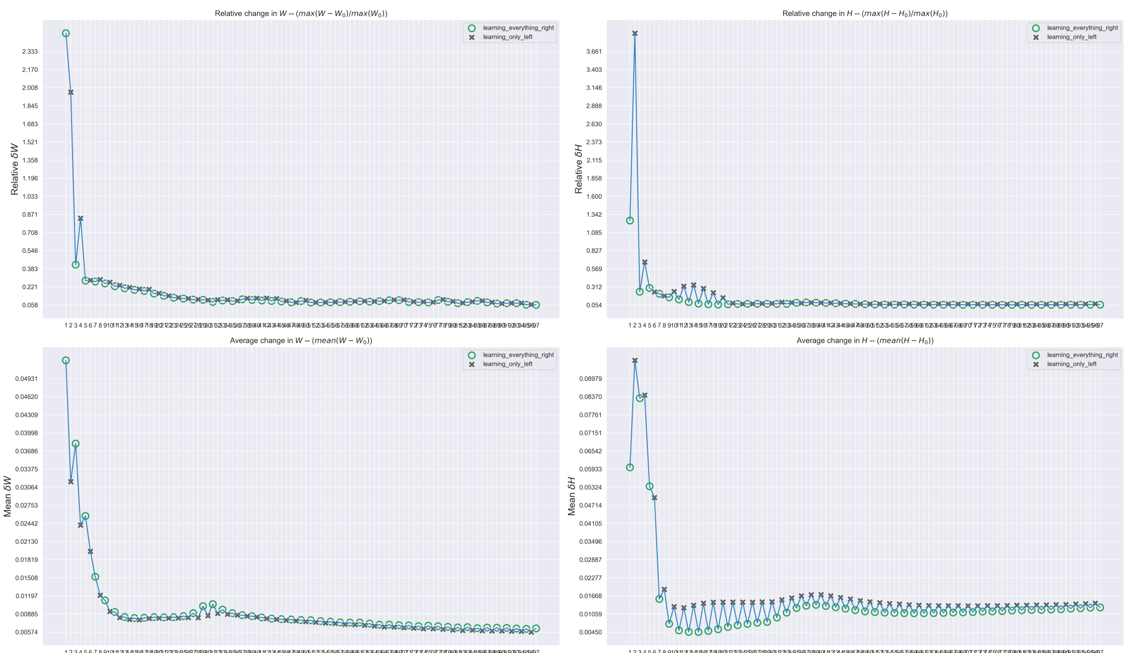


Figure 5.8: Analysis of the change in values of the distributed representations W and H in between epochs. The learning of DCSE-twin for this analysis is the standard one-to-one pace between the left-side and right-side models. The top two figures show the relative change of values, and the bottom two figures show the average change of the values. These figures are from the perspective of DCSE-right, so the green circles represent when DCSE-right is updating the weights of W and H, and the black crosses show when DCSE-right is idle and DCSE-left is updating the values.

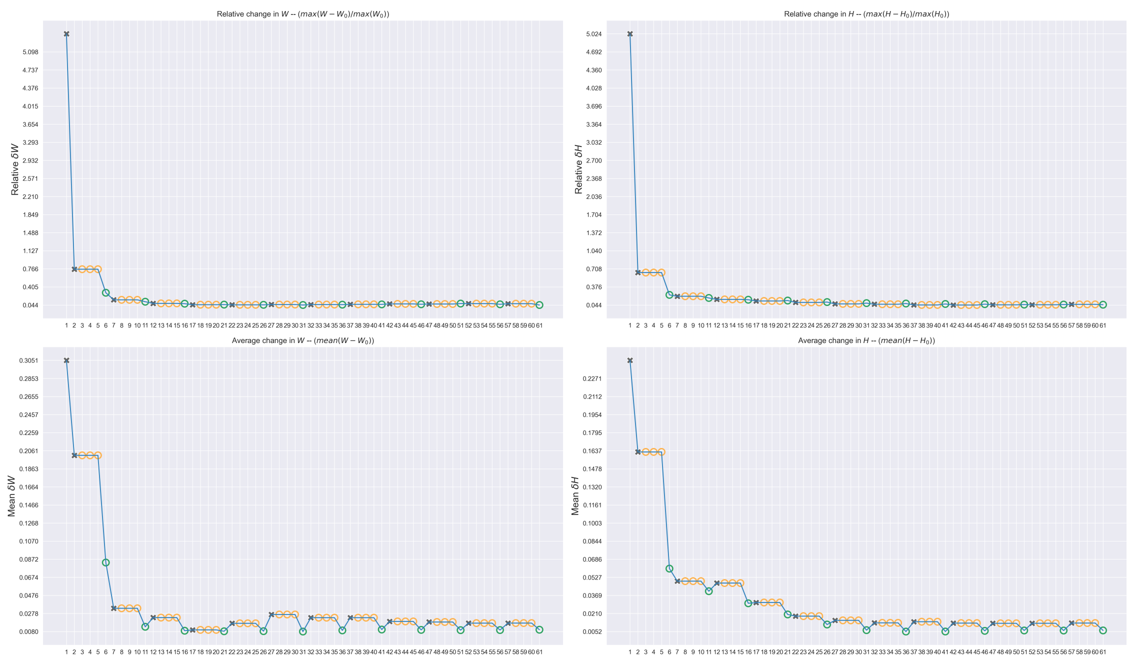


FIGURE 5.9: Analysis of the change in values of the distributed representations W and H in between epochs. The learning of DCSE-twin for this analysis is based on *freezing* the weights of W and H for 3 epochs. The top two figures show the relative change of values, and the bottom two figures show the average change of the values. These figures are from the perspective of DCSE-right. The green circles represent when DCSE-right is updating the weights of W and H, the black crosses show when DCSE-right is idle and DCSE-left is updating the values, and lastly, the yellow circles indicate when the model is only updating the values of the combination layers.

"What I love about science is that as you learn, you don't really get answers. You just get better questions."

John Green

6 Conclusion and future work

In this dissertation, I have addressed the problem of predicting adverse side effects of drug combinations. DCSE-twin is motivated by the concept of learning distributed representations that can describe the biological activity of drugs, drug pairs and side effects. I provided extensive experimental results that show that DCSE-twin outperforms the state-of-the-art under different novel evaluation scenarios that are pharmacologically relevant. My method has great value to be adopted in regular drug development pipelines in the premarketing stages. This will empower the analysis of important safety concerns for investigational drugs at earlier stages, leading to more efficient triage for novel drug candidates in the development process [24]. Methods, such as DCSE-twin, don't necessarily prove causality association, but instead, they can be valuable tools to flag hypotheses of adverse DDIs and support signal assessment. A comprehensive clinical judgement will always prove to be fundamental for establishing causal relationships between drugs and safety concerns, like side effects.

6.1 Future work

Interpretability of the distributed representations

The black-box nature of machine learning methods is one of the major limitations leading to difficulty in interpreting what features are important or interpreting the model's predictions [24]. Interpretability can bridge the gap between applied machine learning research and clinical practice. Having biological interpretations of machine learning predictions can help scientists involved in high-stake decisions to make informed decisions. To this end, a clear objective for my future work is to address the interpretability of the distributed representations of the drugs, drug pairs and side effects.

From the results of the reproducibility analysis, presented in Chapter 5.5, it is clear that some further tuning is needed to make the representations reproducible across multiple runs. Analysing the different sizes for the distributed representations may improve the model's overall reproducibility. Following the reproducibility of the representations, I plan to adopt a similar procedure as described in the work by Galeano et al. [44], to evaluate the interpretability of the representations. This analysis has the potential of interpreting specific components of the representations by relating them to different anatomical activities of drugs and specific drug routes of administration.

Predicting side effects for higher-order drug combinations

Another avenue for future work involves the prediction of side effects for higher-order (i.e. more than two) drug combinations. It has been recently reported [67], that on average, the elderly population, take two to nine prescription medications per day. A different survey by Kaufman et al. [68] found that 57% of women aged ≥ 65 years took ≥ 5 prescription medications and 12% took ≥ 10 medications. In line with these results, a large study of drug therapy in elderly home care patients in Europe [69] ($n = 2707$; mean age 82.2 years) found that 51% of patients took ≥ 6 medications per day and 22% of patients ≥ 9 drugs per day. It has also been reported that $5 - 15\%$ of elderly patients suffer clinically significant adverse reactions due to drug interactions, whereas the number of elderly patients exposed to a potential DDI is estimated to be between 35 and 60% [70].

Recently, the NSIDES [55] dataset, which includes millions of higher-order interactions of drugs, became available. This dataset allows for the deployment of computational methods that can assist in the prediction of side effects associated with higher-order drug combinations.

An important advantage of DCSE-twin is that its architecture allows for a straightforward extension to more than just two drugs. This is achieved simply by adding more drug inputs to the model. It would involve concatenating more individual drug distributed representations in Eq. 4.2 to learn a representation for the higher-order combination. The changes to the rest of the model should be very minimal. An important note is that the dimension of the representation does not need to increase as we increase the number of drugs in the combination. As a consequence, the overall computational complexity of DCSE-twin will be largely independent of the number of drugs being considered. Moreover, the training phase of the higher-order interaction models could be initialised using the weights of already trained lower-order models, leading to faster learning and better performance. This is in stark contrast with existing methods for DDI prediction, in which the input is normally a concatenation of drug features [4, 71] into very high-dimensional vectors. Extending these methods for higher-order combinations would require input vectors of increasingly high dimensions and a complete re-training of the entire model.

A Appendix

A.1 DCSE-right CNN variation

The choice of nonlinearity for DCSE-right is an MLP. However, this can be changed to any other form of nonlinearity as long as the output is a representation for the drug pair that lies in the same latent space as the representations for the individual drugs.

Here I present a variation of DCSE-right in which I use a Convolutional Neural Network (CNN) to combine the representations of the individual drugs. As unorthodox as it may seem to apply a CNN to a setting where the input data is not necessarily “matrix” shaped, like an image, for example, it is still possible to make use of the properties of CNNs for data like ours. Inspired by [72], instead of concatenating the representations for the individual drugs, I perform the outer product between them, as follows:

$$O^{i,j} = \vec{w}_i \vec{w}_j^T, \tag{A.1}$$

where $O^{i,j} \in \mathbb{R}^{k,k}$ is a square matrix that represents the outer product between the representations for drugs i and j. From here, we can use $O^{i,j}$ as the input for a CNN to learn features from the outer product between the representations. The idea is to use the outer product to explicitly model the pairwise correlations between the elements of the individual drug representations in the latent space. This matrix $O^{i,j}$ is called the Interaction Map. An overview of this nonlinear combination can be observed in Figure A.1.

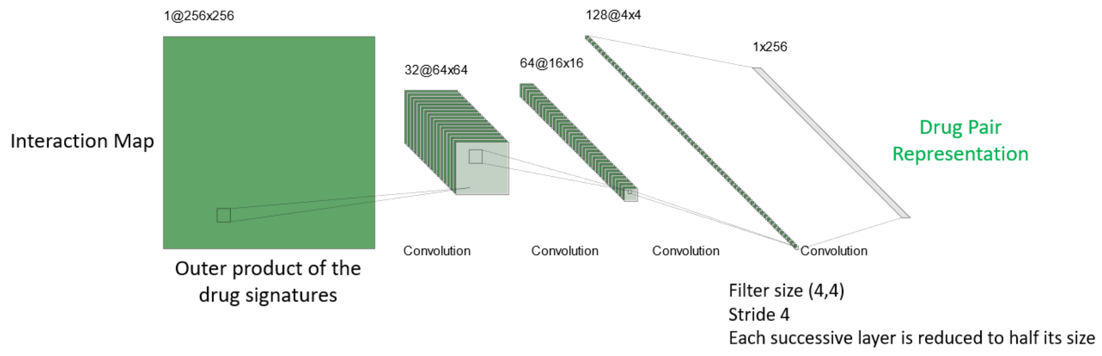


Figure A.1: CNN variation for DCSE-twin. The interaction map represents the outer product between the individual drug representations, multiple convolution layers are stacked on top of the Interaction Map. The final layer represents the distributed representation for the drug pair.

Such rich semantics in the interaction map facilitate the following nonlinear layers to learn possible high-order dimension correlations. Moreover, the matrix form of the interaction map makes it feasible to learn the interaction function with the effective CNN, which potentially is able to generalise better and can go deeper than the fully connected MLP. Another advantage of the CNN over the MLP is the large number of parameters that the MLP needs to learn. Besides that, a simple concatenation of the representations that serves as input to the MLP, may not be representative enough as the outer product that may be able to model the pairwise correlations between the representations.

To analyse the performance of this variation, I follow the procedure detailed in Chapter 5.1.2. Table A.1 shows the results when comparing the MLP variation, named DCSE-MLP, against this CNN variation, named DCSE-CNN. As observed, DCSE-MLP achieves a higher performance. However, the difference is not that significant, which indicates that this approach is also able to get reasonable predictive performance.

Table A.1: Performance comparison of DCSE-right variations of nonlinear combination, DCSE-MLP and DCSE-CNN.

Method	AUROC	AUPRC
DCSE-CNN	0.880	0.840
DCSE-twin	**0.892**	**0.873**

A.2 Analysis of the learning of DCSE-twin

DCSE-twin is an approach that simultaneously learns distributed representations for drugs and side effects from two different datasets, namely, TWOSIDES and OFFSIDES. The goal of DCSE-twin is to have both models benefiting from each other to obtain better predictions and more robust representations. In order to accomplish this, the training of DCSE-twin needs balance out many different factors that, at times, can be opposing forces during the learning. These factors include the learning algorithm between DCSE-left and DCSE-right, the regularisation parameters, and the modelling of the biases in the two models. In this section I explore all of these factors and present different experiments that contributed to finding the right balance for my final approach.

A.2.1 Learning from two different datasets

In order to leverage the data from OFFSIDES correctly to learn better distributed representations for drugs and side effects, the learning process needs to be done in the right way. The learning of my method is done in an iterative process, as shown in Algorithm 3, where the connection between the left-side and right-side models is in the sharing of the distributed representations and biases. Both models are trying to learn from different data and, although related, different problems. This will generate an inevitable struggle between them. However, the idea is to make this struggle as minimum as possible and have both models benefit from each other and from both datasets. To accomplish this, it is necessary to understand how much the learning of one model is being affected by the other. I analyse the behaviour of the loss functions of the two models during the training to find the right balance for the learning process. Figure A.2 shows a visualisation of this process with the plot of the loss for each model at each epoch. Each column in the figure, that goes through both models, has only one green circle that shows which model is updating the weights. The black crosses show the value of loss for the model, calculated with the weights that were updated by the other model. We can observe that there is a natural tug-of-war motion between the two loss functions. When one model goes through an epoch, the loss for that model decreases (green circle), but the loss for the other model increases (black cross). Nevertheless, the overall loss between two epochs of the same model seems to always be decreasing, which is the desirable behaviour.

Although a simple one-to-one iterative process seems to be heading in the right direction, the fight between the two models is still present. As a way to decrease this struggle during the learning, I came up with an approach in which I modify some aspects during training. First, notice in Figure A.2 how in the early epochs the loss for DCSE-right decreases even when DCSE-left is performing the updates. This suggests that the representations learned by the left side are a good starting point for the training. To make the best of this situation, I start the training with some repeated iterations where only DCSE-left updates the representation. In the early epochs of Figure A.3 we can observe how the loss of DCSE-right goes down when the left side is performing the initial updates. The second change I make is to *freeze* the weights of the representations in DCSE-right for some iterations. Specifically, I want to make sure that DCSE-right is focusing on updating the weights for the MLP to learn a better representation for the drug pair, as opposed to keeping updating the weights for the distributed representations for the single drugs. I adjust Algorithm 3 to *freeze* the weights of the distributed representations of the individual drugs after a certain amount of epochs, this process is shown in Figure A.3. During these epochs, DCSE-right will only update the parameters for the MLP and the representation of the combination of drugs, as depicted by the yellow circles. The loss gets stabilised during the epochs in which the weights are frozen. During these *frozen* epochs, the loss from DCSE-left remains the same, as there is no update to the distributed representations, shown by the red crosses in the figure. However, the struggle between the two models is also decreasing and the loss for both models keeps decreasing at their respective turns to update the weights. Empirically, the number of *frozen* epochs was set to 3 for the final training of DCSE-twin.

There are still some aspects of the learning that can be improved and understood. For instance, the overall loss of the two models gets to a slightly lower value in the simple one-to-one training than in the approach with the *frozen* weights. Nevertheless, the struggle between the two models is minimal in the model where the weights are *frozen*, and this is something I want to prioritise that may help in the reproducibility of the approach. If the two models struggle too much, the overall change of the values of the representations between two iterations is going to be higher. The less the two models fight each other, the absolute value of the updates between two epochs for the representations will also decrease. If the values of the representations change too much between epochs, then the less likely will be that they will be reproducible across independent runs of my approach. This is why keeping the struggle between the two models may benefit not only the learning but also the overall interpretability of the model.

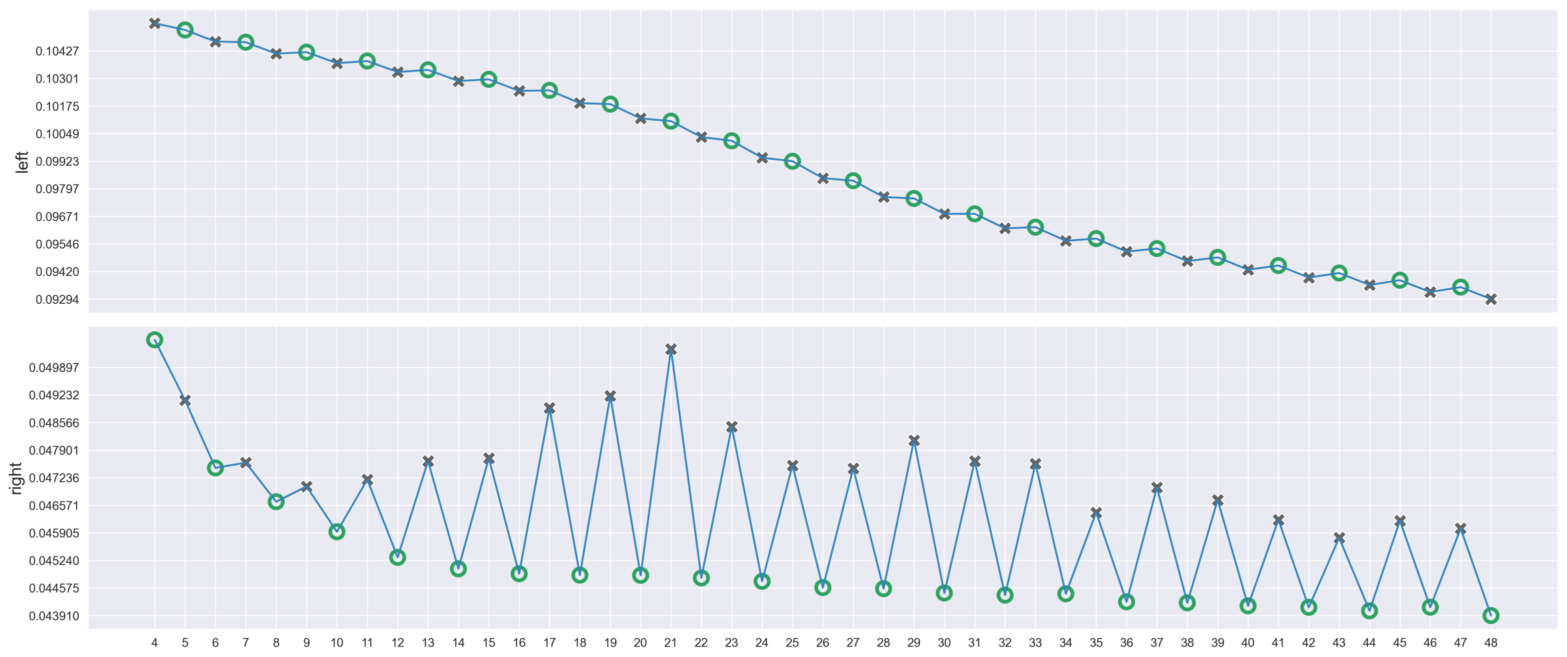


Figure A.2: Learning process of the DCSE-twin. Top subplot is the loss of the DCSE-left and the bottom subplot is the loss of DCSE-right. y-axes show the loss values, x-axes show the epochs. The plots show the iterative process described in Algorithm 3 where the training is alternated at a pace of one epoch per model. The green circles denote which model's turn it is to update the weights. The black crosses show the value of the loss function calculated with the updated weights of the other model. Note that there is only one green circle per column.

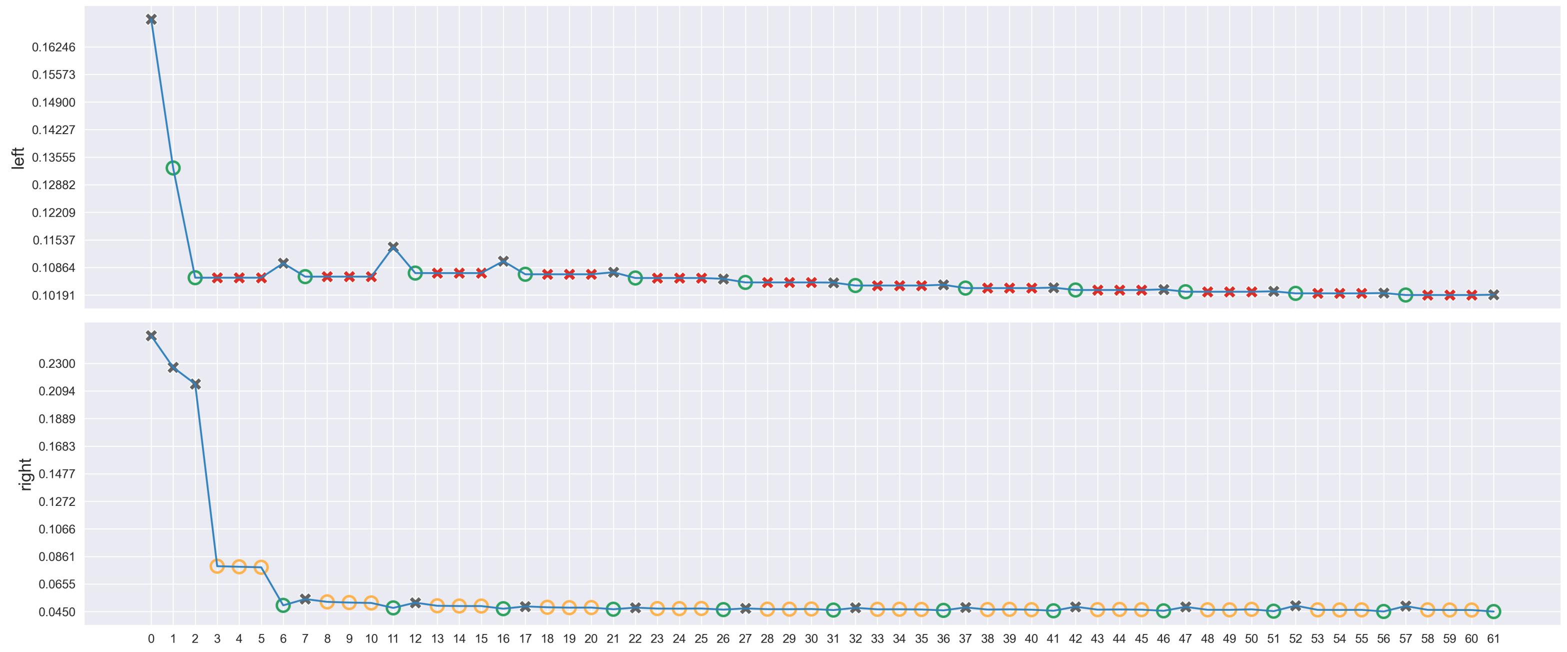
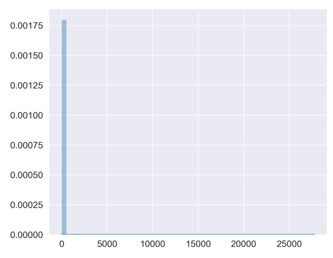


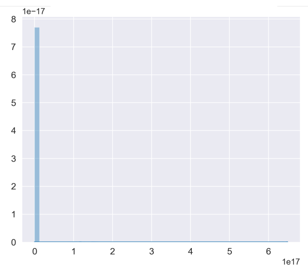
Figure A.3: Learning process of the DCSE-twin when *freezing* the weights for the distributed representations. Top subplot is the loss of the DCSE-left and the bottom subplot is the loss of DCSE-right. y-axes show the loss values, x-axes show the epochs. The plots show the iterative process when *freezing* the weights of the distributed representations in DCSE-right. The yellow circles represent the epochs in which the DCSE-right is only updating the MLP. The red crosses show the loss of DCSE-left with the same weights since there was no update done by DCSE-right.

A.2.2 Regularisation analysis for DCSE-left

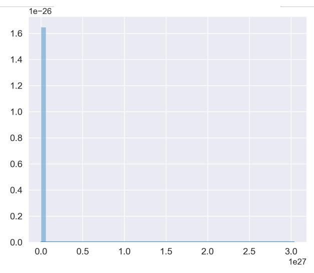
Besides modelling the inherent tendencies in the data, the biases in DCSE-twin have the role to shift the distribution of the non-normalised prediction scores to capture the entire sigmoid range. In Equation 4.7 we can observe how this happens. Note that the dot product between the distributed representation of the drug and the side effect gets subtracted by the product of the drug and side effect biases. This means that the values in u and v need to be closely related to the values in W and H, as these need to be adjusted accordingly to shift the distribution. To understand this relationship between the biases and the distributed representations, I analyse the distribution of their values. Figure A.4 shows these distributions for a model without any regularisation, i.e. β,$\alpha = 0$. As observed in Figs. A.4a, A.4b and A.4c, the values for W, H, and uv^T either are zero or explode in size. The unnormalised scores (Fig. A.4d) are mostly zero which causes the final predictions (Fig. A.4e) to be mostly random at 0.5. These results suggest that for DCSE-left to work correctly, I need to apply some form of regularisation to the distributed representations and also to the biases.



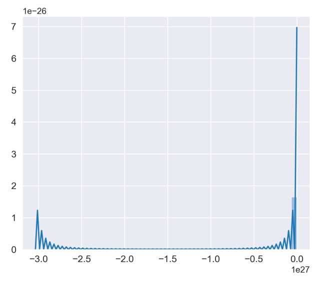
(a) Distribution of W



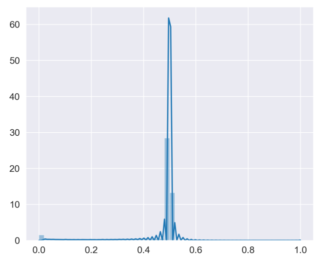
(b) Distribution of H



(c) Distribution of the biases uv^T



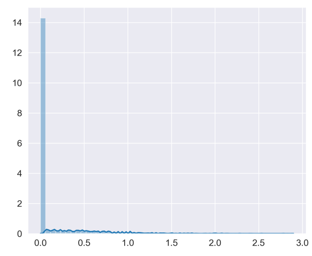
(d) Distribution of the logits $WH - uv^T$

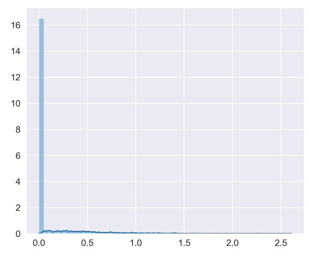


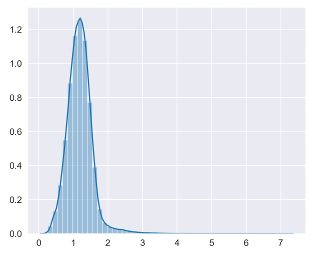
(e) Distribution of the normalised scores $\sigma(WH - uv^T)$

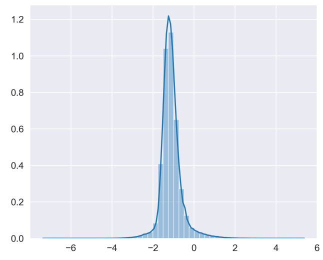
FIGURE A.4: Analysis of the distribution of the distributed representations of the drugs **(a)** and of the side effects **(b)**, the biases **(c)**, the logits or unnormalised scores **(d)**, and the normalised scores **(e)** with no regularisation — $\beta = 0$, $\alpha = 0$.

The right balance between the regularisation hyperparameters, β for W and H, and α for uv^T, is needed to keep the values of the distributed representations and biases from exploding in size. Figure A.5 shows the same distributions as before, but now with regularisation, specifically $\beta = 2$ and $\alpha = 0,1$. After applying this regularisation, the values for W, H, and uv^T are shrunk accordingly, as observed in Figs. A.5a, A.5b and A.5c. From my empirical results, the regularisation for W and H needs to be higher than for u and v. The fact that the regularisation for the distributed representations is higher, is constraining the biases because of how tightly linked they are. If the values for W and H don't grow as much, then there is no need for the biases to grow either, meaning in a lower regularisation for uv^T. Another reason why β should be higher than α is that besides helping capture the range of the sigmoid, u and v also need to model the drugs' and side effects' biases. This allows the distributed representations to learn from more meaningful interactions. Since the regularisation for the biases is lower, they have more freedom to grow to adjust to these two important roles they play in the model.


(a) Distribution of W


(b) Distribution of H


(c) Distribution of the biases uv^T


(d) Distribution of the logits $WH - uv^T$

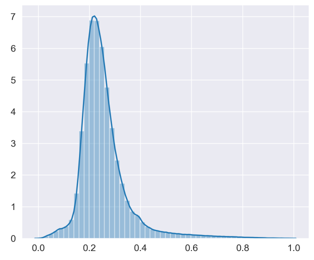

(e) Distribution of the normalised scores $\sigma(WH - uv^T)$

FIGURE A.5: Analysis of the distribution of the distributed representations of the drugs **(a)** and of the side effects **(b)**, the biases **(c)**, the logits or unnormalised scores **(d)**, and the normalised scores **(e)** with the right regularisation — $\beta = 2$, $\alpha = 0.1$.

References

[1] M. Zitnik, M. Agrawal, and J. Leskovec, “Modeling polypharmacy side effects with graph convolutional networks,” *Bioinformatics*, vol. 34, no. 13, 2018. xv, xvi, 2, 14, 15, 16, 18, 26, 61, 74, 78, 81, 91, 94

[2] B. Perozzi, R. Al-Rfou, and S. Skiena, “DeepWalk: online learning of social representations,” in *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD ’14, (New York, NY, USA), pp. 701–710, Association for Computing Machinery, Aug. 2014. xvi, 23, 24

[3] V. Nováček and S. K. Mohamed, “Predicting Polypharmacy Side-effects Using Knowledge Graph Embeddings,” *AMIA Summits on Translational Science Proceedings*, vol. 2020, pp. 449–458, May 2020. xvii, 5, 27, 29, 33, 35, 78, 91

[4] J. Y. Ryu, H. U. Kim, and S. Y. Lee, “Deep learning improves prediction of drug–drug and drug–food interactions,” *Proceedings of the National Academy of Sciences*, vol. 115, pp. E4304–E4311, May 2018. xvii, xviii, 3, 30, 31, 32, 33, 110

[5] K. Han, E. E. Jeng, G. T. Hess, D. W. Morgens, A. Li, and M. C. Bassik, “Synergistic drug combinations for cancer identified in a crispr screen for pairwise genetic interactions,” *Nature biotechnology*, vol. 35, no. 5, pp. 463–474, 2017. 1, 2

[6] M. Bansal, J. Yang, C. Karan, M. Menden, J. Costello, and H. Tang, “A community computational challenge to predict the activity of pairs of compounds,” *Nature Biotechnology*, vol. Dec;32(12):1213–22, 2014. 1, 5

[7] H. Huang, P. Zhang, X. A. Qu, P. Sanseau, and L. Yang, “Systematic prediction of drug combinations based on clinical side-effects,” *Scientific reports*, vol. 4, p. 7160, 2014. 1

[8] J. Jia, F. Zhu, X. Ma, Z. W. Cao, Y. X. Li, and Y. Z. Chen, “Mechanisms of drug combinations: interaction and network perspectives,” *Nature reviews Drug discovery*, vol. 8, no. 2, pp. 111–128, 2009. 1

[9] L. Magro, U. Moretti, and R. Leone, "Epidemiology and characteristics of adverse drug reactions caused by drug–drug interactions," *Expert opinion on drug safety*, vol. 11, no. 1, pp. 83–94, 2012. 2, 3

[10] T. T. Ashburn and K. B. Thor, "Drug repositioning: identifying and developing new uses for existing drugs," *Nature reviews Drug discovery*, vol. 3, no. 8, pp. 673–683, 2004. 2

[11] R. Pan, V. Ruvolo, H. Mu, J. D. Leverson, G. Nichols, J. C. Reed, M. Konopleva, and M. Andreeff, "Synthetic lethality of combined bcl-2 inhibition and p53 activation in aml: mechanisms and superior antileukemic efficacy," *Cancer cell*, vol. 32, no. 6, pp. 748–760, 2017. 2

[12] P. G. Kremsner and S. Krishna, "Antimalarial combinations," *The Lancet*, vol. 364, no. 9430, pp. 285–294, 2004. 2

[13] G. Xiong, Z. Yang, J. Yi, N. Wang, L. Wang, H. Zhu, C. Wu, A. Lu, X. Chen, S. Liu, T. Hou, and D. Cao, "DDInter: an online drug–drug interaction database towards improving clinical decision-making and patient safety," *Nucleic Acids Research*, vol. 50, pp. D1200–D1207, Jan. 2022. 2

[14] D. M. Qato, J. Wilder, L. P. Schumm, V. Gillet, and G. C. Alexander, "Changes in prescription and over-the-counter medication and dietary supplement use among older adults in the united states, 2005 vs 2011," *JAMA internal medicine*, vol. 176, no. 4, pp. 473–482, 2016. 2, 44

[15] M. Pirmohamed and M. Orme, "Drug interactions of clinical importance," *Davies's textbook of adverse drug reactions*, pp. 888–912, 1998. 2

[16] B. Jin, H. Yang, C. Xiao, P. Zhang, X. Wei, and F. Wang, "Multitask dyadic prediction and its application in prediction of adverse drug-drug interaction," in *Thirty-First AAAI Conference on Artificial Intelligence*, 2017. 2

[17] R. I. Patel and R. D. Beckett, "Evaluation of resources for analyzing drug interactions," *Journal of the Medical Library Association : JMLA*, vol. 104, pp. 290–295, Oct. 2016. 2, 4

[18] A. Tornio, A. M. Filppula, M. Niemi, and J. T. Backman, "Clinical Studies on Drug–Drug Interactions Involving Metabolism and Transport: Methodology, Pitfalls, and Interpretation," *Clinical Pharmacology and Therapeutics*, vol. 105, pp. 1345–1361, June 2019. 4

[19] I. R. Edwards and J. K. Aronson, "Adverse drug reactions: definitions, diagnosis, and management," *The lancet*, vol. 356, no. 9237, pp. 1255–1259, 2000. 4

[20] C. G. Hartford, K. S. Petchel, H. Mickail, S. Perez-Gutthann, M. McHale, J. M. Grana, and P. Marquez, "Pharmacovigilance during the pre-approval phases," *Drug safety*, vol. 29, no. 8, pp. 657–673, 2006. 4

[21] R. W. Sanson-Fisher, B. Bonevski, L. W. Green, and C. D'Este, "Limitations of the randomized controlled trial in evaluating population-based health interventions," *American journal of preventive medicine*, vol. 33, no. 2, pp. 155–161, 2007. 5

[22] W. K. Amery, "Why there is a need for pharmacovigilance," *Pharmacoepidemiology and drug safety*, vol. 8, no. 1, pp. 61–64, 1999. 5

[23] N. P. Tatonetti, P. P. Ye, R. Daneshjou, and R. B. Altman, "Data-Driven Prediction of Drug Effects and Interactions," *Science Translational Medicine*, Mar. 2012. Publisher: American Association for the Advancement of Science. 5, 6, 14, 38

[24] H. Ibrahim, A. Abdo, A. M. El Kerdawy, and A. S. Eldin, "Signal Detection in Pharmacovigilance: A Review of Informatics-driven Approaches for the Discovery of Drug-Drug Interaction Signals in Different Data Sources," *Artificial Intelligence in the Life Sciences*, vol. 1, p. 100005, Dec. 2021. 6, 92, 108

[25] G. E. Hinton, "Distributed representations," 1984. 8

[26] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed Representations of Words and Phrases and their Compositionality," in *Advances in Neural Information Processing Systems 26* (C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, eds.), pp. 3111–3119, Curran Associates, Inc., 2013. 8, 22, 25, 61, 78

[27] G. E. Hinton *et al.*, "Learning distributed representations of concepts," in *Proceedings of the eighth annual conference of the cognitive science society*, vol. 1, p. 12, Amherst, MA, 1986. 8

[28] D. E. Rumelhart, G. E. Hintont, and R. J. Williams, "Learning representations by back-propagating errors," p. 4, 1986. 8

[29] C. Knox, V. Law, T. Jewison, P. Liu, S. Ly, and A. Frolkis, "DrugBank 3.0: a comprehensive resource for 'omics' research on drugs," *Nucleic acids research*, vol. 39, no. suppl_1, 2010. 14

[30] P. Zhang, F. Wang, J. Hu, and R. Sorrentino, "Label Propagation Prediction of Drug-Drug Interactions Based on Clinical Side Effects," *Scientific Reports*, vol. 5, p. 12339, July 2015. 14

[31] A. Kastrin, P. Ferk, and B. Leskošek, "Predicting potential drug-drug interactions on topological and semantic similarity features using statistical learning," *PLOS ONE*, vol. 13, p. e0196865, May 2018. 14

[32] S. Vilar, E. Uriarte, L. Santana, C. Friedman, and N. P Tatonetti, "State of the art and development of a drug-drug interaction large scale predictor based on 3d pharmacophoric similarity," *Current Drug Metabolism*, vol. 15, no. 5, pp. 490–501, 2014. 14

[33] S. Vilar, R. Harpaz, E. Uriarte, L. Santana, R. Rabadan, and C. Friedman, "Drug—drug interaction through molecular structure similarity analysis," *Journal of the American Medical Informatics Association*, vol. 19, no. 6, pp. 1066–1074, 2012. 14

[34] J.-Y. Shi, J.-X. Li, K. Gao, P. Lei, and S.-M. Yiu, "Predicting combinative drug pairs towards realistic screening via integrating heterogeneous features," *BMC Bioinformatics*, vol. 18, p. 409, Oct. 2017. 14

[35] S. Vilar Varela, E. Uriarte, L. Santana, N. P. Tatonetti, and C. Friedman, "Detection of drug-drug interactions by modeling interaction profile fingerprints," 2013. 14

[36] A. Cami, S. Manzi, A. Arnold, and B. Y. Reis, "Pharmacointeraction network models predict unknown drug-drug interactions," *PloS one*, vol. 8, no. 4, p. e61468, 2013. 14

[37] T. Trouillon, J. Welbl, and S. Riedel, "Complex Embeddings for Simple Link Prediction," p. 10. 22

[38] D. Weininger, "Smiles, a chemical language and information system. 1. introduction to methodology and encoding rules," *Journal of chemical information and computer sciences*, vol. 28, no. 1, pp. 31–36, 1988. 30

[39] X. Chu, Y. Lin, Y. Wang, L. Wang, J. Wang, and J. Gao, "Mlrda: A multi-task semi-supervised learning framework for drug-drug interaction prediction," in *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, pp. 4518–4524, AAAI Press, 2019. 33

[40] J. Zhu, Y. Liu, and C. Wen, "MTMA: Multi-task multi-attribute learning for the prediction of adverse drug–drug interaction," *Knowledge-Based Systems*, vol. 199, p. 105978, July 2020. 33

[41] N. Xu, P. Wang, L. Chen, J. Tao, and J. Zhao, "Mr-gnn: Multi-resolution and dual graph neural network for predicting structured entity interactions," *arXiv preprint arXiv:1905.09558*, 2019. 33

[42] A. Deac, Y.-H. Huang, P. Veličković, P. Liò, and J. Tang, "Drug-Drug Adverse Effect Prediction with Graph Co-Attention," *arXiv:1905.00534 [cs, q-bio, stat]*, May 2019. arXiv: 1905.00534. 33

[43] S. Qian, S. Liang, and H. Yu, "Leveraging genetic interactions for adverse drug-drug interaction prediction," *PLOS Computational Biology*, vol. 15, p. e1007068, May 2019. 34, 89

[44] D. Galeano, S. Li, M. Gerstein, and A. Paccanaro, "Predicting the Frequency of Drug Side effects," *Nature communications*, 2020. 37, 39, 49, 58, 109

[45] M. Kuhn, I. Letunic, L. J. Jensen, and P. Bork, "The sider database of drugs and side effects," *Nucleic acids research*, vol. 44, no. D1, pp. D1075–D1079, 2016. 39

[46] D. D. Lee and H. S. Seung, "Learning the parts of objects by non-negative matrix factorization," *Nature*, vol. 401, pp. 788–791, Oct. 1999. 42

[47] D. D. Lee and H. S. Seung, "Algorithms for Non-negative Matrix Factorization," in *Advances in Neural Information Processing Systems 13* (T. K. Leen, T. G. Dietterich, and V. Tresp, eds.), pp. 556–562, MIT Press, 2001. 42, 54, 63, 67, 102

[48] G. Hinton, "Deep learning—a technology with the potential to transform health care," *Jama*, vol. 320, no. 11, pp. 1101–1102, 2018. 46

[49] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, and T.-S. Chua, "Neural Collaborative Filtering," in *Proceedings of the 26th International Conference on World Wide Web*, WWW '17, (Perth, Australia), pp. 173–182, International World Wide Web Conferences Steering Committee, Apr. 2017. 53

[50] P. Chandak and N. P. Tatonetti, "Using Machine Learning to Identify Adverse Drug Effects Posing Increased Risk to Women," *Patterns*, vol. 1, p. 100108, Oct. 2020. 57

[51] D. P. Kingma, D. J. Rezende, S. Mohamed, and M. Welling, "Semi-Supervised Learning with Deep Generative Models," *arXiv:1406.5298 [cs, stat]*, Oct. 2014. arXiv: 1406.5298. 61

[52] J. S. Larsen and L. K. H. Clemmensen, "Non-negative Matrix Factorization for binary data," in *2015 7th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K)*, vol. 01, pp. 555–563, Nov. 2015. 63, 64

[53] Z. Zhang, T. Li, C. Ding, and X. Zhang, “Binary matrix factorization with applications,” in *Seventh IEEE international conference on data mining (ICDM 2007)*, pp. 391–400, IEEE, 2007. 64

[54] N. P. Tatonetti, P. P. Ye, R. Daneshjou, and R. B. Altman, “Data-Driven Prediction of Drug Effects and Interactions,” *Science Translational Medicine*, vol. 4, p. 125ra31, Mar. 2012. 75

[55] “tatonetti-lab/nsides-release.” 77, 109

[56] B. Malone, A. García-Durán, and M. Niepert, “Knowledge Graph Completion to Predict Polypharmacy Side Effects,” in *Data Integration in the Life Sciences* (S. Auer and M.-E. Vidal, eds.), Lecture Notes in Computer Science, (Cham), pp. 144–149, Springer International Publishing, 2019. 78

[57] A. Bordes, N. Usunier, A. Garcia-Duran, J. Weston, and O. Yakhnenko, “Translating Embeddings for Modeling Multi-relational Data,” p. 9. 80

[58] L. Zhang, Y. D. Zhang, P. Zhao, and S.-M. Huang, “Predicting drug–drug interactions: an fda perspective,” *The AAPS journal*, vol. 11, no. 2, pp. 300–306, 2009. 86

[59] A. Cami, A. Arnold, S. Manzi, and B. Reis, “Predicting Adverse Drug Events Using Pharmacological Network Models,” *Science Translational Medicine*, vol. 21;3(114):114ra127-114ra127, Dec. 2011. 86

[60] N. Raj, S. Fernandes, N. R. Charyulu, A. Dubey, R. GS, and S. Hebbar, “Post-market surveillance: a review on key aspects and measures on the effective functioning in the context of the united kingdom and canada,” *Therapeutic advances in drug safety*, vol. 10, p. 2042098619865413, 2019. 91

[61] P. Zhang, F. Wang, J. Hu, and R. Sorrentino, “Label propagation prediction of drug-drug interactions based on clinical side effects,” *Scientific reports*, vol. 5, p. 12339, 2015. 91

[62] R. Girshick, J. Donahue, T. Darrell, and J. Malik, “Rich feature hierarchies for accurate object detection and semantic segmentation,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 580–587, 2014. 96

[63] L. Alexandrov, S. Nik-Zainal, D. Wedge, P. Campbell, and M. Stratton, “Deciphering Signatures of Mutational Processes Operative in Human Cancer,” *Cell Reports*, vol. 3, pp. 246–259, Jan. 2013. 100

[64] L. B. Alexandrov, S. Nik-Zainal, D. C. Wedge, S. A. J. R. Aparicio, S. Behjati, A. V. Biankin, G. R. Bignell, N. Bolli, A. Borg, A.-L. Børresen-Dale, S. Boyault, B. Burkhardt, A. P. Butler, C. Caldas, H. R. Davies, C. Desmedt, R. Eils, J. E. Eyfjörd, J. A. Foekens, M. Greaves, F. Hosoda, B. Hutter, T. Ilicic, S. Imbeaud, M. Imielinski, N. Jäger, D. T. W. Jones, D. Jones, S. Knappskog, M. Kool, S. R. Lakhani, C. López-Otín, S. Martin, N. C. Munshi, H. Nakamura, P. A. Northcott, M. Pajic, E. Papaemmanuil, A. Paradiso, J. V. Pearson, X. S. Puente, K. Raine, M. Ramakrishna, A. L. Richardson, J. Richter, P. Rosenstiel, M. Schlesner, T. N. Schumacher, P. N. Span, J. W. Teague, Y. Totoki, A. N. J. Tutt, R. Valdés-Mas, M. M. van Buuren, L. van 't Veer, A. Vincent-Salomon, N. Waddell, L. R. Yates, J. Zucman-Rossi, P. Andrew Futreal, U. McDermott, P. Lichter, M. Meyerson, S. M. Grimmond, R. Siebert, E. Campo, T. Shibata, S. M. Pfister, P. J. Campbell, and M. R. Stratton, "Signatures of mutational processes in human cancer," *Nature*, vol. 500, pp. 415–421, Aug. 2013. Bandiera_abtest: a Cg_type: Nature Research Journals Number: 7463 Primary_atype: Research Publisher: Nature Publishing Group Subject_term: Cancer Subject_term_id: cancer. 100

[65] S. Vassilvitskii and D. Arthur, "k-means++: The advantages of careful seeding," in *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*, pp. 1027–1035, 2006. 101

[66] L. Kaufman and P. J. Rousseeuw, *Finding groups in data: an introduction to cluster analysis*. John Wiley & Sons, 2009. 101

[67] E. R. Hajjar, A. C. Cafiero, and J. T. Hanlon, "Polypharmacy in elderly patients," *The American journal of geriatric pharmacotherapy*, vol. 5, no. 4, pp. 345–351, 2007. 109

[68] D. W. Kaufman, J. P. Kelly, L. Rosenberg, T. E. Anderson, and A. A. Mitchell, "Recent patterns of medication use in the ambulatory adult population of the united states: the slone survey," *Jama*, vol. 287, no. 3, pp. 337–344, 2002. 109

[69] D. Fialová, E. Topinková, G. Gambassi, H. Finne-Soveri, P. V. Jónsson, I. Carpenter, M. Schroll, G. Onder, L. W. Sørbye, C. Wagner, *et al.*, "Potentially inappropriate medication use among elderly home care patients in europe," *Jama*, vol. 293, no. 11, pp. 1348–1358, 2005. 109

[70] J. Doucet, P. Chassagne, C. Trivalle, I. Landrin, M. Pauty, N. Kadri, J. Ménard, and E. Bercoff, "Drug-drug interactions related to hospital admissions in older adults: a prospective study of 1000 patients," *Journal of the American Geriatrics Society*, vol. 44, no. 8, pp. 944–948, 1996. 109

[71] A. Kastrin, P. Ferk, and B. Leskošek, “Predicting potential drug-drug interactions on topological and semantic similarity features using statistical learning,” *PloS one*, vol. 13, no. 5, p. e0196865, 2018. 110

[72] X. He, X. Du, X. Wang, F. Tian, J. Tang, and T.-S. Chua, “Outer Product-based Neural Collaborative Filtering,” *arXiv:1808.03912 [cs, stat]*, Aug. 2018. arXiv: 1808.03912. i