DRUG-TARGET INTERACTION PREDICTION USING IMBALANCE AWARE MULTI-LABEL METHODS

Dimitris Iliadis
AEM:763

Supervisor
Grigorios Tsoumakas, Assistant Professor

April 2, 2019
Abstract

Every year, the pharmaceutical industry faces the challenge of increasing the efficiency of developing new, effective and inexpensive drugs. Over the last few decades, machine learning methods took advantage of the large datasets available to big pharma and became a viable alternative for the development of new drugs. The aim of the present thesis was to experiment with different imbalance aware, multi-label methods on drug-target interaction datasets. The methods we implemented include the classic Binary Relevance approach, the Ensemble of Classifier Chains with Random Undersampling (ECCRU) and the Multi-Label Co-Training method. We experimented with two different datasets, one with a considerable size, comparable to what the pharmaceutical companies use in their day-to-day operations and a much smaller dataset that is widely used by other researchers in the area of drug-target interaction prediction (gold standard datasets). We also examined different problem settings and showed that some can be considered quite unrealistic and falsely boost the performance of methods that are used in the area of drug-target interaction prediction.
Περίληψη

Κάθε χρόνο, οι φαρμακευτικές εταιρείες αντιμετωπίζουν την απαιτητική πρόκληση να αυξήσουν την αποτελεσματικότητα της διαδικασίας ανάπτυξης νέων, αποδοτικών και φθηνών φαρμάκων. Τις τελευταίες δεκαετίες, μεθόδοι μηχανικής μάθησης έχουν εκμεταλλευτεί τα μεγάλα σύνολα δεδομένων που ήταν διαθέσιμα στις μεγάλες φαρμακευτικές εταιρείες και έχουν φτάσει στο σημείο να αποτελούν μια βιώσιμη εναλλακτική λύση για την ανάπτυξη νέων φαρμάκων. Σκοπός της παρούσας διπλωματικής εργασίας ήταν ο πειραματισμός με διαφορετικές μεθόδους πολλαπλών ετικετών που λαμβάνουν υπόψη την ανισορροπία των κλάσεων, πάνω στο πρόβλημα της πρόβλεψης της αλληλεπίδρασης φαρμάκου-βιολογικού στόχου. Οι μέθοδοι που εφαρμόσαμε περιλαμβάνουν την κλασική προσέγγιση της δυαδικής συνάφειας, το σύνολο των ταξινομητών αλυσίδας με τυχαία υποδειγματοληψία και τη μέθοδο συν-μάθησης πολλαπλών ετικετών. Τα πειράματα μας έγιναν πάνω σε δύο διαφορετικά σύνολα δεδομένων, ένα με μέγεθος που μπορεί να συγκριθεί με αυτό που χρησιμοποιούν οι φαρμακευτικές εταιρείες στα ερευνητικά τους τμήματα και ένα πολύ μικρότερο σύνολο δεδομένων που χρησιμοποιείται ευρέως από ερευνητές στον τομέα της πρόβλεψης αλληλεπίδρασης φαρμάκου-βιολογικού στόχου. Επίσης εξετάσαμε διαφορετικές προσεγγίσεις του προβλήματος της πρόβλεψης αλληλεπίδρασης φαρμάκων-βιολογικών στόχων και δείχνουμε ότι κάποιες προσεγγίσεις μπορούν να αποδειχθούν μη ρεαλιστικές και τελικά να ενισχύουν με εσφαλμένο τρόπο την απόδοση των μεθόδων που χρησιμοποιούνται σε αυτόν τον τομέα.
3.2.3 Ensembles of Classifier Chains with Random Undersampling (ECCRU) .................. 44
3.2.4 Multi-Label Co-Training (MLCT) ................................. 47

4 Results .................................................. 53
4.1 Performance Metrics .............................................. 53
4.2 Gold Standard Datasets Results .............................. 55
4.3 ChEMBL Dataset results ........................................... 60

5 Future Work .............................................. 63
Chapter 1

Introduction

Every year, the pharmaceutical industry faces the challenge of increasing the efficiency of developing new, effective and inexpensive drugs. What is observed, however, is that every year fewer and fewer new drugs become available on the market (Scannell et al., 2012). Machine learning methods can take advantage of the large data sets available to major pharmaceutical companies and therefore become an alternative to the development of new drugs.

The first step in the design of a drug is to identify a biomolecular target on which a potential drug can act, e.g. a protein whose activity can be modified by a chemical compound to achieve a beneficial therapeutic effect. The next step is to sort tens of thousands of chemical compounds with high-performance bioassays that record interactions with the target. The final goal of this procedure is the identification of a new lead compound that interacts with the desired protein target. The chemical structure, scaffold, is modified to optimize efficiency and reduce possible side effects. High-throughput screening measures produce large amounts of data that can be used to train machine learning models.

In addition, there are many protein targets for which high-performance testing is not available, so their screening is particularly time-consuming and costly. At this point in the drug discovery process, biological screening can be replaced by virtual screening, i.e., computational predictions for interactions between the biological target and the chemical compound. However, in order for this approach to be accepted by pharmaceutical companies, the accuracy of the prediction should be high. Another important criterion that determines the success of the virtual screening method is its ability to detect new scaffolds that interact with the target.

A key advantage of virtual screening is that it does not limit the testing of a compound’s interaction with only one target but allows for simultaneous screening with multiple targets proteins at once. Typically, chemical compounds which show high affinity with many proteins result in many undesirable side effects. Also, binding to other proteins than the primary target leads to a reduction in the drug’s
bioavailability which then can lead to reduced efficacy. These are the main causes of the failure of many candidate drugs in the clinical trials phase. Failures due to undetected side effects are particularly time-consuming, costly and are considered the main reason for reduced efficacy in the discovery of new drugs. The use of virtual screening allows the hierarchy of candidate drugs based on the affinity on multiple target proteins, which can reduce the risk of drug discovery.

Approaches to virtual screening, such as target prediction, can be grouped into structure-based and ligand-based. Structure-based methods simulate the physical interactions between a compound and a biomolecular target (Kitchen et al., 2004), but can only be applied if the full three-dimensional structure of all the interacting molecules is known. Also, they can not be applied to large databases due to their high complexity. Ligand-based approaches predict the activity of a compound to a biomolecular target based on previous measurements recorded for it (Kurczyk et al., 2015). Methods based on machine learning are mainly used in ligand-based approaches. Typical examples are Naive Bayes (Xia et al., 2004; Abdo et al., 2010), SVM (Han et al., 2008; Sengupta and Bandyopadhyay, 2014; Jorissen and Gilson, 2008) and neural networks (Morro et al., 2018; Hagan and Hagan, 2016).

Pharmaceutical companies often use commercial software such as SEA (Keiser et al., 2007), SwissTargetPrediction (Gfeller et al., 2013), CSNAP (Lo et al., 2015), and others.

Based on the success of machine learning techniques, we evaluate the performance of ensembles of classifier chain models with random undersampling EC-CRU and semi-supervised multi-label methods and compare them with modern methods and commercial solutions in different problem settings. To that end, we experimented with the widely used in the drug-target interaction prediction problem, gold standard datasets as well as a dataset based on the ChEMBL chemical database, a database that has many similarities to internal databases maintained by large pharmaceutical companies, although much smaller. The ChEMBL database contains over 1,200 biological targets, 1.3 million chemical compounds and 13 million characteristics based on ECFP12 fingerprints.
Chapter 2

Background

2.1 Biological Background

Here we go through some basic biological terms of molecular biology that will be used in the following section:

• Antigen: Characterizes a complex molecule that when it enters the body, the immune system produces antibodies against it. Each antigen has distinct surface features, called epitopes that can result in specific responses.

• Antibody: A Y-shaped protein with a unique binding site shape which locks onto the distinct shape of an antigen. Antibodies are produced from lymphocytes (a type of white blood cells) as a response to the detection of a pathogen in the body.

• Proteomics: A large-scale study of proteins.

• Alleles: A variation of a gene (one member of a pair) that is located at a specific location of a particular chromosome and that can lead to the expression of different traits.

2.1.1 Proteins

Proteins are large macromolecules that can be found in abundance in living systems and that show great diversity in structure and function compared to other classes of macromolecules. A single cell can contain thousands of different proteins with each one expressing a different function. Despite the considerable variation in their structures and functions, all proteins are consisting of one or more long chains of amino acids. The following are two main types of proteins that play important roles in the normal function of a cell or an organism:
Enzymes: are macromolecules that catalyse biochemical reactions by lowering the activation energy. They initiate the reaction, speed up (in a rate of millions or trillions times faster than uncatalyzed) the reaction process and also verify the result is always the same. The essential tasks of life, such as metabolism, proteins synthesis, cell renewal, and growth are all regulated by enzymes. The molecules that serve as a starting material for reactions and which are acted upon by enzymes are called substrates. The enzyme converts the substrates into different molecules known as products.

Hormones: are signaling molecules released by endocrine cells and transported by the circulatory system. They target distant organs to control specific physiological processes, like digestion, metabolism, growth and development, movement, reproduction, and mood. Hormones bind to a specific receptor of the target cell which is also a protein made by that cell. When the hormone binds to the receptor, it causes a change in the receptor’s shape. This change allows the receptor to fit with other cell molecules which then triggers new activities in the cell. Hormones can be categorized based on their chemical structures. Peptide hormones like insulin are chains of amino acids. Steroid hormones are lipids (fat-like molecules) whose structures are derived from cholesterol. Monoamine hormones like adrenaline and noradrenaline are made by modifying amino acids.

Other protein functions: In the respiratory system, hemoglobin carries substances like oxygen throughout the body in blood or lymph. Antibodies or immunoglobins protect the body by helping to recognize and destroy foreign pathogens in the immune system. Proteins like actin and tubulin help to build cellular structures, while keratin provides structural support to dead cells that become fingernails and hair. Finally, actin and myosin carry out muscle contraction, while albumin provides nutrients for the early development stages of an embryo.

2.1.2 Amino Acids

Proteinogenic amino acids are the structural units (monomers) that make up proteins. From the 500 naturally occurring amino acids, only 20 contribute to the creation of proteins. 10 out of those 20 amino acids are considered vital for humans because the body cannot synthesize them and they must be obtained from the diet. A protein is made up of one or more linear chains of amino acids, each of which is called a polypeptide. Except for proteins, amino acids can form shorter polymer chains called peptides. Amino acids share a basic structure, which consists of a central carbon atom (alpha-α carbon) bonded to an amino group $NH_2$, a
carboxyl group \( COOH \), and a hydrogen atom. The differentiating factor between
the 20 amino acids is the R group they contain. It is easy to conclude that there
are 20 different R groups that determine the identity of the amino acid as well as
its properties. For instance, if the R group is a hydrogen atom, the amino acid
is glycine. However, if the R group is a methyl \( (CH_3) \), the amino acid is an ala-
nine. The properties of the side chain also determine the amino acid’s is considered
acidic, basic, polar (hydrophile) or nonpolar (hydrophobe).

Hydrophobicity is how much an amino acid wants to avoid water, i.e. it would
be unstable when in contact with a water molecule. Therefore, a sequence of amino
acids in the cell will fold into a specific 3D structure to minimizes its energy. Hy-
drophobic or non-polar, amino acids tend to be located deep inside a protein while a
hydrophilic will be more likely on the surface. Amino acids with opposite charges
will tend to be close to each other while ones with same charges are likely to be
farther away. The size of the amino acid also puts a constraint on the possible
configuration of the protein. In general, a protein will have a certain 3D structure
based on its amino acid sequence, something that is very difficult to be computa-
tionaly predicted based on the sequence alone.

2.1.3 DNA and Protein Production

Cells are fundamental units of living organisms. The genetic information of a per-
son is stored in and replicated through DNA or deoxyribonucleic acid, which is
located inside the nucleus. A molecule of DNA consists of two chains that are
wrapped around each other, forming a double helix shape. Each chain is made up
of subunits, called nucleotides. There are four different types of nucleotides that
differ based on the type of the base that is present. These four different nucle-
obases are, adenine \( (A) \), cytosine \( (C) \), guanine \( (G) \), thymine \( (T) \). The two chains
are bonded together as a result of chemical bonds that create pairs of bases from
different chains. The bases that participate in these bonds cannot be random. Ade-
nine forms a base pair with thymine that is held together by two hydrogen bonds,
while guanine pairs with cytosine with three hydrogen bonds.

The relationship between DNA and proteins is one that is vital to sustaining
life. DNA codes for proteins and in turn, these proteins have an multiple func-
tions as state in the above sections. The first step that occurs is a process of
protein coding is known as transcription. During this phase DNA is converted
to messenger RNA (mRNA) by an enzyme called RNA polymerase. RNA is a
molecule that is chemically very similar to DNA. The basic difference is that in
RNA the nucleobase thymine \( (T) \) is replace by uracil \( (U) \). The conversion of DNA
to \( mRNA \) occurs when an RNA polymerase makes a complementary messenger
RNA \( (mRNA) \) copy of a DNA sequence. Once the \( mRNA \) molecule has been
created, specific chemical modifications are made to enable the next step, called
During translation mRNA is decoded in a ribosome to produce a specific amino acid chain, or polypeptide that later folds into an active protein and performs its functions in the cell. A group of three *mRNA* nucleotides encodes a specific amino acid, called a codon. The beginning and end of translation is signaled by two of the codons, called start and stop codons. The final protein product is formed after the stop codon has been reached.

![Figure 2.1: 20 different amino acids](image)

**2.2 Machine Learning Background**

Machine learning is a subfield of computer science, and its primary goal is to build programs that have the potential to learn without any explicit programming. As a research field, machine learning has addressed many problems and has proposed several methods to tackle them. The main factors of differentiation are the knowledge representation as well as the possible existence of initial knowledge and data that could be used for the training of the model.
As a branch of Artificial Intelligence (AI), machine learning deals with the study of algorithms that improve their behavior in assigned tasks by creating models and templates from a set of data. With regard to the design of machine learning systems in the field of symbolic AI, learning is defined as the ability to acquire additional knowledge that changes the volume or the characteristics of the existing knowledge base. On the other hand, in the field of non-symbolic AI, as in Artificial Neural Networks, learning is characterized as the ability to modify the internal structure of the model.

Although present systems have a long way to go to reach the multidimensional learning ability of humans (General AI), machine learning has displayed huge advancements. Specific algorithms have been particularly successful in specific areas (Narrow AI) and thus have allowed the emergence of modern applications (e.g., spam filtering, AlphaGo) with great success.

### 2.2.1 Types of Machine Learning

The field of machine learning can be divided into three key learning subcategories: supervised learning, unsupervised learning, and reinforcement learning. More specifically:

- **Supervised Learning** is the process of mapping an output based on training data, that is, a set of examples that are used for model training. In supervised learning, each example has an input (typically a vector of attributes or features) and the desired output value. Supervised learning algorithms analyze training data and produce a model that can be used to describe new examples.

- **Unsupervised Learning** is the process of discovering associations and groups that may hide behind unlabeled data, based only on their properties. This method produces patterns with each one describing partially the data. This can be achieved with methods like association rules and clustering.

- **Reinforcement Learning** is a process in which the learning mechanism tries to learn a strategy of action through its direct interaction with the environment. The logic of reinforcement learning is inspired by the learning models of living beings in which reward and punishment are used. It is mainly used for design problems, such as robot movement control and the optimization of work pipelines in factories.
2.2.2 Supervised Learning Algorithms

Inductive learning algorithms are considered a subtype of learning algorithms in the field of machine learning. The purpose of these algorithms is to derive decisions, relevant to the relationships that dominate the set of training examples that have been gathered. Supervised Inductive Learning algorithms are mainly applied to classification problems in which discrete values are predicted and regression problems in which numerical values are predicted. A variation of the approach above (semi-supervised machine learning algorithms) is used in forecasting problems. In this case, the algorithms operate with training set, in which there are examples with unknown outputs.

In Supervised Induction Learning, the system aims to construct a model in the form of a prediction function, which will assign given inputs to known, desirable outputs. The ultimate goal is to generalize the function for inputs with unknown outputs. The function has the following characteristics:

- The function input is called an instance.
- These instances are described by their attributes.
- The training set is a sub-set of all the instances.
- The rest of the instances form the test set and are used to determine the performance of the trained model.
- The target function is a function that maps an input to a known output.
- In supervised learning, the predictive performance of the target function can be improved by introducing an error function that can measure the 'distance' of the predicted output to the actual output.

2.2.3 Single label Learning

Instance-based learning

The storage and use of specific samples can improve the performance of multiple supervised learning algorithms. These include decision trees, classification rules, and distributed networks. The instance-based learning algorithm does not utilize a training step. The training data set remains unused and is only accessed when there is a need to classify a new instance. The sorting of a new instance is based on its similarity to the stored data set. The similarity check by these (lazy) algorithms is done by using a distance function that is customized to the data. Such algorithms do not create an internal model that represents the knowledge acquired
by the training set but identifies a target function locally. These approaches shift the majority of the run time from the training stage to the classification stage.

**Decision Trees**

Decision Trees is a popular Supervised Learning algorithm, which was first introduced by Quinlan (1986) and has been successfully applied in many areas such as face recognition, medical diagnosis, and knowledge mining in general. The decision tree extraction algorithm leads to the creation of a tree structure describing the data graphically. The leaves of this structure are the classes and the root-to-leaf paths represent classification rules. Decision trees are constructed using only attributes that can distinguish the classes. Training data are used to construct the tree and test data to calculate its accuracy. A significant advantage of this method is that each prediction can be attributed to a sequence of easily understandable decisions (one decision per node). It usually does not require any prior statistical knowledge to read and understand them.

**Support Vector Machines (SVM)**

SVMs were introduced by Hearst (1998) and have become a very popular classification technique. Each SVM essentially attempts to find a boundary or a partition separating the input data into two classes. The separating hyperplane is positioned using examples from the training set that are known as support vectors. When the data can not be separated linearly, kernel functions can be used to transform the feature space into higher dimensions. At these higher dimensions there is a greater probability that the data are linearly separable. Samples from the test set are mapped to the same feature space and their class is predicted according to the side of the hyperplane they end up. The distance from the threshold can be used to set the predictions confidence level. The larger the distance, the higher the confidence in the prediction. A major drawback with classifiers like SVMs as well as neural networks is that they are considered black box methods (in contrast to decision trees), making it difficult to interpret the results.

**Neural Networks (NN)**

Neural networks evolved from research into computer models of the brain. The two most commonly used neural network architectures in computational chemistry research are feed-forward networks and the Kohonen network.

- **Feed-forward networks**: A neural network of this type consists of multiple layers of nodes with connections between all the pairs formed by the nodes
of the adjacent layers. Each node can be found in zero or one state. The status of each node changes depending on the nodes of the previous level with which they are connected as well as the weights of these connections. A key feature of the feed forward neural network is the presence of one or more layers of hidden nodes. This feature, combined with the backpropagation algorithm that is used to train them, has enabled the successful implementation of neural networks in various ways, such as Computer vision, Speech Recognition, Medical Diagnostics, etc.

The neural network must first be trained. This is achieved by using a set of training, ie a set of instances as well as their respective outputs. Weights and other parameters in the neural network initially have random values, so the initial output of the network (from one or multiple output nodes) differs significantly from the desired outputs. Thus, a vital phase of the training phase involves modifying the various weights and parameters that govern the state of the nodes and their connections. The now trained neural network, can now be used to predict the values for new samples. The feed-forward network is a supervised learning method, because it uses the values of dependent variables to create its model.

- **Kohonen networks**: The kohonen network or self-organizing map \cite{kohonen2012} is an unsupervised learning method. A network of this type usually consists of a rectangular array of nodes. Each node has a relative vector with small random values that corresponds to the input data. For each instance presented as an input to the network, the distance from each
node vector is calculated using the following distance metric:

\[ d = \sum_{i=1}^{p} (x_i - u_i)^2 \]  

(2.1)

where \( u_i \) is the value of the i-th component of the vector u for that node, \( x_i \) is the corresponding value for the input vector and \( p \) is the number of descriptor’s characteristics. In the end, the node that has the minimum distance from the input vector can be identified so that the node’s weight can be updated in the following way:

\[ u'_i = u_i + \eta (x_i - u_i) \]  

(2.2)

where \( \eta \) is the profit factor. A key feature of the Kohonen network is that, in addition to updating the winning node, its neighbors are also informed using the same equation. This updating mechanism identifies the winning node as well as its neighbors that are more similar to the input. This results in the creation of areas with common nodes. During training, the profit rate gradually decreases, thus slowing down the rate at which the nodes change.

**Logistic regression**

Logistic regression is a model for the classification of the values of a \( Y \) variable based on probability theory. In this application, the dependent variable \( Y \) has a binary character (it can only have two values, active and inactive) and the goal is to predict its outcome from a number of independent variables that may be numerical or categorical. The study of the relationship of the categorical dependent variable cannot be realized through the standard Linear Regression algorithm because predicting a categorical variable practically means calculating the probability with which the dependent variable will receive the specific value. The probability value should be between 0 and 1, something that Linear Regression cannot because it calculates probability values outside these limits.

Logistic regression is used to predict the probability of occurrence of an event by adjusting the input data to the logistic curve as shown in Figure 2.3.

Binary logistic regression is a binomial equation in which the response variable \( Y \) is the random result of one of two possible outcomes (success or failure) (e.g. the result of a coin roll). The binary logistic regression has the following form:
In the above function, $z$ is the input variable and $f(z)$ its result. The main characteristic that leads to its use is that while the input variable receives positive and negative values, the result of $f(z)$ is limited between 0 and 1. The $z$ variable is the result of the action of a group of independent variables while $f(z)$ determines the probability of a particular result due to the action of this group. The variable represents the measure of the total contribution of all participating independent variables in the model.

\[ f(z) = \frac{e^z}{1 + e^z} = \frac{1}{1 + e^{-z}} \]  \hspace{1cm} (2.3)

Figure 2.3: sigmoid function

\[ z = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_k X_k \]  \hspace{1cm} (2.4)

\begin{itemize}
  \item $\beta_0$ is the height of the slope of the regression line. Equals the $z$ value when all values of independent variables are 0.
  \item $\beta_i$ are the coefficients of regression, each of which expresses the amount of contribution of the corresponding variable. A positive value of the coefficient indicates that this variable increases the probability of a successful outcome while a negative value means that it decreases that probability.
\end{itemize}
Naive Bayes classifiers include a set of classifiers that are based on applying the Bayes theorem while also making the assumption of strong independence between the features. This approach is especially suited when the dimensionality of the inputs is high. Despite its simplicity, it often outperforms more advanced classification methods like support vector machines. Naive Bayes was first introduced in the text retrieval field and it remains in use in categories like spam detection that use word frequencies as features. This method can be scaled very easily because it requires a number of parameters linear in the number of features. It can be used in binary or multiclass classification settings and it provides different types of algorithms like GaussianNB, MultinomialNB, and BernoulliNB. The basic disadvantage of the algorithm lies in the assumption that all the features are unrelated. This assumption makes the algorithm able to learn the importance of individual features but at the same time unable to discover important relationships between these features.

To better understand the naive Bayes classifier we have to explain in more detail the Bayes theorem. The Bayes theorem works on conditional probability, which is the probability that something will happen, given that something different has already happened. Using this notion, we can use prior knowledge to calculate the probability of an event. The formula for calculating the conditional probability is presented below:

$$P(H|E) = \frac{P(E|H) \times P(H)}{P(E)}$$  \hspace{1cm} (2.5)

where

- $P(H)$ and $P(E)$: are the probabilities of $H$ and $E$ being true independently of each other (known prior probability or marginal probability)
- $P(H|E)$: is the probability of $H$ being true, given that $E$ is true
- $P(E|H)$: is the probability of $E$ being true, given that $H$ is true

The naive Bayes classifier uses the Bayes theorem to predict the probabilities for each class such as the probability that a given instance belongs to a specific class. Then the class with the highest probability is selected as the most likely class (Maximum A Posteriori or MAP).
Genetic Algorithms

This is a category of algorithms in which the approximation of the target function starts with a set of hypotheses represented as bit sequences. This set is constantly changing with the use of mutation and crossover processes, which are directly borrowed from biology. In each iteration, a generation of hypotheses arises from its past, by merging past features and mutating them into new ones. Each step of this process involves choosing a subset with the most healthy assumptions, which will form the basis for the formation of a new generation. Genetic algorithms are mainly used in optimization problems.

2.2.4 Multi-Target Prediction

Traditional methods in machine learning and statistics provide data-based models for predicting one-dimensional targets, such as binary outputs in classification and numerical outputs in regression. In recent years, the need for applications in diverse fields has led to the development of basic research into more complex problems requiring multiple target predictions. Such problems arise in various fields, such as document categorization, label recommendations in images, videos and music, drug discovery, etc.

More specifically, a characteristic that can differentiate the approaches other than the type of predicted labels is their number. Both single-label classification and regression have to predict only one label, while in multi-target prediction the model has to predict multiple labels at once. In particular, multi-label classification predicts multiple categorical targets, while multivariate regression predicts multiple numerical targets. These approaches can be applied to a large number of different problems, which by their very nature require the prediction of their multiple labels.

The relations and interdependencies that exist between the targets can have a great effect on the prediction outcome. More specifically, the prediction of multiple targets has many properties that have led to the development of many variations like:

- The targets type: Every target can be binary, numerical, based on some order as well as some combination them.
- The type of prediction: The prediction can be direct or indirect. For example in a simple multi-target classification setting the classifier returns every target’s confidence value while in a label ranking setting we assume that we only know about the ranking between the targets.
- The amount of features that every instance has: The instances features can be the same of have significant differences.
• The targets attributes can be present or not.

• The training instances can be the same or they can display differences.

• There might be a need to make predictions for all targets, a subset of all targets of even only one target.

Multi-Label Classification

Several different algorithms have been proposed in recent years to address the problem of multi-target classification. These algorithms are divided into two groups, the algorithm adaptation methods, and the problem transformation methods. In the first class of methods, the goal is to extend algorithms that until then, were used to classify single targets, so that they can adapt to the need for multi-label data management. In the second class, the algorithms remain unchanged, as the purpose of these methods is to transform the classification problem itself into multiple single classification problems. In this approach, simple classifiers can be easily used without any adaptation.

Problem transformation methods

There are several transformations that can lead to the transformation of a set of multi-labeled data into a new data set in which each example is matched to a single label.

• **Binary Relevance method**: The Binary Relevance (BR) transformation method is based on the one-vs-all method, which is used for multiple class classification problems. In the BR method, the original data set is transformed into individual smaller sets where each one only predicts one label and excludes the rest. In order to classify a new instance, it is necessary to combine the predictors of the individual classifiers. Although this method displays good results in many sorting problems, it has a very significant drawback. It cannot, by its very design, exploit possible associations between labels and naively assumes that they are independent.

• **Label Powerset method**: The Label Powerset (LP) transformation method converts the multi-label problem into a multi-class single-label classification problem where the possible values for the transformed attribute of the class are the sum of the unique discrete subsets of the labels present in the original training data. By using this method, learning from multiple-label examples is equivalent to finding a mapping from the attribute space to the label space. The LP method uses the relations that appear between labels.
CHAPTER 2. BACKGROUND

However, when a large or even moderate number of labels is used, the process of learning multiple labels using all possible sets of labels becomes a very challenging task due to the large (exponential) number of these possible sets. The characteristic mentioned above is considered a significant disadvantage of the method. Another critical problem with this method is the imbalanced distribution of classes, which occurs when there are classes in the training set represented by a small number of examples.

2.2.5 Semi-supervised learning

Semi-supervised learning is a set of machine learning methods that make use of unlabeled data. These techniques typically combine a small amount of labeled data with a large number of unlabeled examples to make predictions. Semi-supervised learning can be seen as a compromise between unsupervised learning (exclusive use of unlabeled data) and supervised learning (use of fully labeled train set). Nowadays vast amounts of data are generated every second that are mainly unlabeled. Furthermore, the labeling of these datasets often requires skilled humans and usually ends up being a costly and time-consuming procedure. This cost makes supervised learning an impractical option and semi-supervised learning a very intriguing prospect. Various methods have been introduced to accomplish this goal: self-training (Chapelle et al., 2009), co-training (Blum and Mitchell, 1998a), graph methods et al. All these methods attempt to make use of this combined labeled and unlabeled information to exceed the classification performance that could be achieved either by dismissing unlabeled examples in favour of supervised learning or by ignoring the labels and favoring unsupervised learning. For that to happen, some assumptions about the distribution of the dataset are made. These hypotheses include the smoothness of labels with respect to the features, the low density of the decision boundary and the good performance of clustering methods in the presence of only unlabeled data.

Co-training

Co-training is a machine learning algorithm that was introduced by Blum and Mitchell (1998a) that uses only small amounts of labeled data and large amounts of unlabeled data. The basic idea of this method is that every sample can be partitioned into two distinct views and that either one of these views is sufficient for successful learning (conditionally independent). Instead of using these views by themselves, co-training suggests that the use of both views together can allow unlabeled data to augment a significantly smaller set of labeled samples. The algorithm in more details has the following steps. Given a set \( L \) of labeled examples and a set \( U \) of unlabeled examples, the algorithm creates a small pool \( U' \) with \( u \)
unlabeled examples. The next step is the following iteration. First, it uses $L$ for the training of two distinct classifiers $h_1$ and $h_2$. $h_1$ is a classifier based only on the first view of $L$ and $h_2$ a classifier based on the second view of $L$. The two now trained classifiers make predictions for every example in $U'$ and select the $p$ samples most confidently predicted as positive and $n$ samples most confidently predicted as negative. Each one of these selected samples along with the label assignments is added to $L$. Finally, $U'$ is refilled with $2p + 2n$ randomly selected examples from $U$ and then the iteration continues.

2.3 The Drug Design process

The process of marketing a drug is dominated by the total cost of the effort. It is estimated that the average cost of introducing a new drug to the market is around $2 billion.

It is of great importance that the majority of research programs devoted to the discovery of new drugs are focused on diseases that affect a large proportion of the world’s population or are likely to generate significant revenue for the company. Conditions like hypertension, diabetes, cancer, and Alzheimer monopolize the interests of major pharmaceutical companies.

The first step in the drug discovery process is to obtain a deep understanding of the biochemical pathways involved in a disease. This work is mainly carried out by a molecular biology group that belongs to a pharmaceutical company or third party like a research group in a university.

The molecular biology team examines processes that take place inside the cell and tries to understand the detailed mechanisms in which the disease works. They also try to isolate molecules that are found in the body and are responsible for the disease. These biochemical pathways are controlled by molecules called proteins. Every protein that participates in these pathways is a possible point where the research team can intervene. This is the main reason why the molecular biology group selects the protein that plays the key role as the biological target for the drug. The binding of the drug to the biological target influences the biological pathway and eventually the disease’s state inside the body.

After that, the molecular biology team designs and produces a chemical assay, a test that allows the measurement of the binding affinity of the chemical compound to the target receptor protein. This test is a chemical or biological, in-vitro test which returns a positive response only if the appropriate binding agent reacts with the receptor.

Upon completion of the above step, the weight of the program is passed to the pharmaceutical chemistry team. This team starts with the lead discovery process. The chemical test that was previously developed is now used in a mass screening
test where 1 million or more different molecules are tested in a few days or even weeks. During this process, the pharmaceutical company examines its in-house database in its entirety because if an assay is successful, the chemical compound is usually well-characterized and therefore a prototype of the candidate ligand can be produced very quickly. The most important (active) molecules in this preliminary activity tests are marked as successes. Of the 1,000,000 available assays, successful ones are usually in the range of 1,000. Although the search space is drastically reduced, its size is still too large for further control of each candidate individually. For this reason, the molecules are filtered through other tests that detect properties beyond the binding affinity to the target. These properties include the ability to diffuse through cell membranes, interactions with various metabolic processes as well as the possibility of patentability. For example, a candidate compound may achieve a very good binding affinity to the biological target but at the same time a high degree of homogeneity with another molecule that has been licensed by another pharmaceutical company.

The glaring successes are promoted to the next stage and are characterized as lead compounds. Of the 1,000 candidate molecules of the previous stage, only 5-10 will be promoted to the next step as lead candidates.

Then, the whole process moves towards the optimization of the lead compounds. At this stage, the pharmaceutical chemistry group aims to improve the degree of binding affinity that the compound has for the target, but also improve upon pharmacodynamics and in vivo pharmacokinetics. The above goals are achieved with the modification of the lead compound’s structure. The repeated process of modifying the structure based on results obtained from animal experiments provide a fully-optimized lead. For example, it is reasonably possible that a change to the structure of the compound, so that it does not react with other receptors in the body, greatly reduces the likelihood of unexpected side effects. While candidate compounds are routinely tested on animals, toxicity tests are usually carried out only in the last steps of the drug design process. These tests include standardized protocols that aim to determine the safety and effect of a candidate drug to a living organism.

If the results are favorable, all the laboratory and biological data that are collected during the research process are submitted by the pharmaceutical company as an investigational new drug application (IND) to the respective government agency. If the application is accepted, the lead compound is characterized as a new research drug or a clinical candidate.

This is the starting point of the clinical trial’s procedure which is presented in the four separate steps below:

• Phase 1: the first phase trials usually involve a small number of participants (about 10 to 20 individuals) who are usually healthy volunteers. The purpose
of this phase is to determine the immediate safety of the drug as well as the mechanism of absorption, metabolism, and removal from the organism. This stage also moniters various pharmacokinetic information (e.g. half-life of the drug). Since animal toxicity tests have been conducted before the first phase, it is rare for a candidate drug to fail at this stage because of high toxicity levels in the volunteers.

- Phase 2: the second phase trials include volunteers (usually 100 to 200 people) who suffer from the disease. Tests of this phase continue to control safety but are mainly focused on determining the effectiveness and finding the optimal dose that can achieve therapeutic effects. More specifically, at this point, researchers analyze the optimal dose, the optimal dosing schedule as well as the risk associated with possible short-term side effects that may arise. These tests involve volunteers who are taking the manufactured drug, volunteers who use an inactive substance (placebo) and volunteers who receive the most popular drug with similar activity at the time (if anyone exists).

- Phase 3: the third phase of the trials may involve 1,000 or more patients in different parts of the world to control the drug’s behavior in diverse populations. During this phase, critical statistical data on the safety, efficacy and risk-benefit ratio of the investigational drug are being produced. Safety and various other factors for specific subpopulations (e.g. diabetics or children) are also closely monitored during this phase. In addition, this stage is the basis for determining the guidelines that will be printed on the drug’s label (information about possible reactions with other medicines, special dosages, etc.). In this way, the pharmaceutical company can define age limits or different dosages for different population groups. During this phase, the company starts preparations for large-scale production as well as for submitting the application to the relevant state body.

All the results collected from the three phases mentioned above are sent to the respective national organization in the form of a new drug application (NDA). If the application is accepted (after about 1 year), the candidate compound is also an official drug that can be marketed to the public.

- Phase 4: the fourth phase is practically a never-ending follow-up test on the new drug.
CHAPTER 2. BACKGROUND

2.4 The problem of drug-target interaction prediction

Despite the advances in several steps of the drug discovery process pipeline in the past decades, experimental identification of drug-target interactions (DTI) is still a highly costly and time-consuming process. According to Morgan et al. (2011), bringing a new drug from conception to market can often take nearly a decade and can reach an average cost of 1.8 billion US dollars. Additionally, nearly 90 percent of drugs that reach the stage of clinical trials and seek approval from the regulatory agencies fail to achieve acceptable efficacy and toxicity levels as well as appropriate manufacturing standards.

Drug-target interaction prediction is a crucial step in the drug discovery process. Reliable DTI prediction provides a better understanding of the underlying disease mechanism and also helps to detect unwanted side-effects and unexpected, off-target activities in general. As a result, drug-target interaction prediction can also be used effectively in drug repositioning-repurposing. In this approach, the goal is to discover new biological targets and so, new diseases that can be managed by existing drugs. The primary benefit of this method is that the existing drug is already considered safe by regulatory agencies like the Food and Drug Administration (FDA) and has already generated a lot of information about potential long-term side-effects in different demographic populations. One additional example that showcases the importance of accurate drug-target interaction prediction can be found in the emerging concept of polypharmacology. This approach tries to transform the traditional one-target one-disease paradigm by introducing the idea of tracking the interaction of a drug with multiple targets. These targets can be associated with a single disease pathway as well as multiple pathways in more complex diseases.

During the last decades, various in silico approaches have been developed for the prediction of drug-target interaction. These methods include many ligand and/or structure-based approaches like those that try to uncover relations between drugs and targets through quantitative structure-activity relationships (QSARs) or molecular docking. Both of these categories of methods display certain flaws. Ligand-based methods try to predict interactions between a ligand and a target by comparing this new ligand to the known binding ligands of the target protein. However, this method is extensively time-consuming, and the 3D information about the protein’s structure is often unavailable.

On the other hand, target-based methods try to predict the drug-target interaction by heavily relying on the 3D structure information of the target proteins. However, this 3D information about the protein’s structure is often unavailable. This problem has an even more significant impact on proteins like ion channels
and G-protein-coupled receptors because of their increasingly complex structures. The community has tried to overcome the challenges of the above-mentioned approaches by shifting towards a new drug discovery strategy, called chemogenomics. This approach integrates the genomic space of target proteins and the chemical space of drug compounds into a unified framework to infer new drug-target pairs.

These integrative chemogenomic approaches can be grouped into different categories such as learning-based methods, graph-based methods, and network-based methods \cite{Mousavian2014}. Among the chemogenomic approaches, machine learning-based methods have gained the most attention. In this work, we focus on machine learning methods that are used in drug-target interaction prediction.

We also classify the machine learning methods into two major categories such as supervised and semi-supervised methods. Supervised learning methods use both positive and negative labeled examples in the training set. Then these labeled samples are used to train the learning models that are used for DTI prediction. On the other hand, semi-supervised learning methods make predictions only based on a small amount of labeled data and a large amount of unlabeled data.

Drug target prediction methods can also be grouped based on their ability to use available side information about the chemical compounds and target proteins. Some methods utilize only side information about the problems instances, usually in the form of feature representations of the chemical compound structure or a more structured representation such as network or graph. Other methods are able to use additional information about the targets of the problem, typically in the form of feature or graph representations of the proteins.

The latter category of methods is used to more complex problems that are often referred to as dyadic prediction problems (also called as link prediction or network inference). \cite{Pahikkala2014b} made an important distinction between four main settings of the general dyadic prediction problem. Given $d$ and $t$ as the feature representations of the compounds and proteins respectively, four settings can be extracted:

- **Setting A**: $t$ and $d$ are observed during training in different interaction pairs and the $(d,t)$ pair must be predicted.
- **Setting B**: the $(d,t)$ pair must be predicted while $t$ is observed during training in some interaction pair and $d$ is not.
- **Setting C**: the $(d,t)$ pair must be predicted while $d$ is observed during training in some interaction pair and $t$ is not.
- **Setting D**: neither $t$ or $d$ is observed in any interaction pair of the training set and the $(d,t)$ pair must be predicted.
Setting A is the most extensively researched setting amongst them. This setting results in the problem of matrix completion where both instances and targets are partially observed. That means that we don’t have the interaction information for every possible instance-target pair. These missing values are usually predicted by ignoring the side information in the instance and target space and by then applying techniques like matrix factorization.

Settings B and C are quite similar. When there is available side information about the instance and target space, the two settings become interchangeable. This characteristic can lead to the false notion that interchanging the rows and columns (compounds and proteins) will not affect the predictive accuracy. The reality is that usually, one feature representation will lead to better discrimination than the other. This feature is something that many methods take into account.

2.4.1 Supervised Learning Methods

Supervised Learning Methods can be categorized in the two following classes:

- **Similarity-based methods**: Similarity-based methods utilize drug-drug and target-target matrices and try to predict drug-target interactions either by using only these matrices or by combining them with interaction networks/bipartite graphs. Similarity matrices can be used in different kernel functions:
  - Nearest neighbor methods: This type of methods use information about the nearest neighbors of targets to make prediction
  - Bipartite local models: Bipartite local models turn the problem of predicting edges in a bipartite graph into a binary problem. This is usually performed in a two step process. In the first step two models are trained based on the compounds and targets and in the second step, the final prediction about some drug-target pair is calculated based on the two independent prediction scores of the members.
  - Matrix factorization methods: Matrix factorization first learns the latent feature matrices for drugs and targets, and then multiplies these two matrices to reconstruct the initial interaction matrix.

- **Feature vector-based methods**: feature-based methods represent each drug-target pair with a vector of descriptors. Various features like 1D, 2D fingerprints, substructures, functional groups and amino acid sequences can be used to encode different properties of drugs and targets. These feature representations are then fed into machine learning models (Random Forest, Support Vector Machines) for predicting drug-target interactions.
2.4.2 Semi-supervised Learning Methods

A common problem when facing the drug–target interactions prediction problem with supervised methods is the lack of a negative examples. This is because we can not obtain drug–target pairs without interactions. Published experimental research in the drug-target interaction only reports biological experiments that yield positive drug–target interactions. Therefore, the unknown interaction pairs have been regarded as negative samples in the supervised learning setting. Inaccurate negative sample selection is a main factor in the predictive accuracy of the supervised models.

To face the challenge of this selection problem, semi-supervised methods were designed to use few labeled data and many unlabeled data.

2.5 Methods

The general framework of machine learning for predicting drug–target interactions has three stages: (1) preprocess the drug and target input data (2) training a model based in the train set and (3) utilize the trained model to predict drug-target interactions from the test set examples. A key underlying assumption of machine learning methods is that similar drugs tend to share similar targets and vice versa.

In the following sections, we classify machine learning methods into two categories, the supervised learning, and semi-supervised learning methods. Furthermore, the supervised learning methods can be classified into two sub-classes, the similarity-based and feature-based methods.

2.5.1 Similarity-Based Methods

**Nearest Neighbor Method (NNM)**

Yamanishi et al. (2008) proposed a kernel regression method to derive drug-target interactions by integrating the drug’s chemical structure, the protein’s genomic sequence, and the drug-target’s network topology. Their proposed procedure is the following:

- Both drugs and targets in the interaction network are mapped into a unified space, called pharmacological space.

- Use of the chemical, genomic and pharmacological space to learn two regression models ($f_c$ and $f_g$) and map the compound/protein pairs onto the pharmacological space.
• Predict drug-target interaction pairs by computing the closeness between. For a new compound $c_{new}$ and a new protein $g_{new}$ we can map them based on the two models on the pharmacological space

\[ U_{c_{new}} = \sum_{i=1}^{n_c} s_c(c_{new}, c_i) w_{c_i} \]  \hspace{1cm} (2.6)

\[ U_{g_{new}} = \sum_{i=1}^{n_g} s_g(g_{new}, g_i) w_{g_i} \]  \hspace{1cm} (2.7)

where:

- $w_{c_i}$: weight vector for compound $c_i$
- $w_{g_i}$: weight vector for target protein $g_j$
- $s_c$: chemical structure similarity between two compounds
- $s_g$: genomic sequence similarity between two proteins

In this pharmacological space, known drug–target pairs that interact with each other are close. Also, drugs with high structure similarity tend to interact with similar targets and vice versa. Then, potential drug–target interactions are predicted by calculating how close the drugs-target pair is.

**Bipartite Local Models**

Bipartite local models (BLMs) perform two independent sets of predictions, one from the drug side and one from the target side. The final prediction score is then obtained by aggregating the two separate prediction scores.

The BLM approach was first introduced by Bleakley and Yamanishi (2009). The main idea of their work was to predict drug-target interactions by transforming edge prediction problems into binary classification problems. More specifically, a local model is trained for each drug to predict which target would interact or vice versa. In their approach, they used SVM classifiers as local models. The results of the two SVM models were then averaged to produce the final score.

Assuming a drug-target interaction bipartite network, the basic algorithm predicts if there exists an edge $e_{ij}$ (interaction), between a drug $d_i$ and a target protein $t_j$. The algorithm’s steps are the following:

• Exclude target $t_j$ and create a list of all other know targets of $d_i$ (positive samples labeled +1), as well as another list of targets not known to interact with $d_i$ (negative samples labeled -1).
• Train a classifier on the two lists and then use it to predict the label of \( t_j \).

• Exclude drug \( d_i \) and create a list of all other known drugs that target \( t_j \) (positive samples labeled +1), as well as another list with drugs that are not known to interact with \( t_j \) (negative samples labeled -1).

• Train a second classifier on the two lists and then use it to predict the label of \( d_i \).

• Aggregate the predictions from the two classifiers using the \( \max(, ) \) function.

Based in the same approach, \cite{vanLaarhovenetal2011} introduced a drug-target interaction profile method that encodes all interactions. The drug interaction profile encodes the interactions of a drug with all known targets and the target interaction profile does a similar encoding with all the known drugs. These interaction profiles are then used to construct a similarity matrix (kernel), called a Gaussian Interaction Profile Kernel (GIP).

More specifically, the method computes the interaction profile \( y_{d_i} \) of a drug \( d_i \) as a binary vector (1 if there is a known interaction with a target, 0 otherwise). The interaction profile for a target protein \( t_j \) is constructed in a similar way (1 if there is a known interaction with a drug, 0 otherwise). All these drug and target interaction profiles are used as feature vectors by classifiers and are also used for the construction of kernels. The kernel \( K_{GIP,d} \) for drugs \( y_{d_i} \) is constructed in the following way:

\[
K_{GIP,d} = \exp(-\gamma_d|y_{d_i} - y_{d_j}|^2)
\]

(2.8)

where \( \gamma_d \) is the kernel’s width parameter.

The kernel \( K_{GIP,t} \) for the targets \( t_j \) is constructed in a similar way. These GIP kernels can be used with classifiers like the RLS-avg and RLS-kron that were introduced by Laarhoven. The RLS-avg algorithm uses a kernel ridge regression to make predictions. Instead, the RLS-kron utilizes the kronecker product of the two kernels \( K_{GIP,d} \) and \( K_{GIP,t} \) to make predictions for all drug-target pairs at once, something that significantly reduces runtime.

\cite{Meietal2012} presented a new method named BLM-NII that integrated the concept of BLM with Neighbor based Interaction profile Inferring (NII). This approach was introduced as a mean to fix one critical problem of the standard BLM method, which is the inability to train local models for drugs or targets that do not have any known interactions (new drugs or targets).
In the proposed approach, the interaction profiles of the candidate compounds or target proteins are being inferred from the interaction profiles of their neighbors. This interaction profile inferring procedure is defined as follows:

\[ I_j^d(i) = \sum_{h=1}^{N_d} S_{ih} Y_{hj} \]  

(2.9)

where:

- \( N_d \): number of known drugs
- \( S_{ih} \): similarity between drugs \( d_i \) and \( d_j \)
- \( Y_{hj} \): binary value that represents if there is a known interaction between drug \( d_h \) and target \( p_j \)

**Matrix Factorization Methods**

Matrix factorization methods are typically used in recommendation systems. This family of methods became known during the Netflix Prize challenge\(^1\) due to its superior predictive accuracy in finding potential user-item interactions. The problem of drug-target interaction prediction can be regarded as a matrix completion problem.

In this setting, the observed entries in the matrix correspond to known positive interactions between the compounds in the rows and the targets in the columns and the goal is to predict additional positive interactions from the large set of missing entries. A crucial assumption that is made in this problem is that the majority of the missing values correspond to negative interactions, and only a few positive interactions need to be predicted.

Gönen (2012) proposed the Kernelized Bayesian Matrix Factorization with twin kernels (KBMF2K) method for predicting drug-target interactions. This novel Bayesian formulation combines kernel-based nonlinear dimensionality reduction, matrix factorization as well as binary classification. The basic idea behind the approach can be translated into two steps. The first step is to project the drug and target spaces into two low dimensional spaces \( (G_d \) and \( G_t) \) using only chemical similarity for the drug-drug kernel \( (S_d) \) and genomic similarity for the target-target kernel \( (S_t) \) The second step consists of estimating drug-target interactions under those new low dimensional spaces by computing the drug–target interaction score matrix \( F \) as follows:

\[ F = G_d^T G_t \]  

(2.10)

\(^1\)http://www.netflixprize.com
At the time of its proposal, this was the first fully probabilistic formulation for the prediction of drug-target interactions. Despite its originality, the KBMF2K is considered inefficient, because it has to estimate the matrices by an iterative process which starts with randomly selected initial values.

Cobanoglu et al. (2013) proposed an algorithmic methodology based on Probabilistic Matrix Factorization (PMF) that can be used for large datasets. The difference compared to KBMF2K is that PMF does not depend on chemical and genomic kernels for the prediction of drug-target interactions, so no chemical/target similarity information is used and all the unknown interactions are treated as negative instances. When large data sets with lots of known interactions are used, PMF is able to construct better models. On the other hand, when the number of known interactions is limited, the use of the additional information that drug-drug and target-target kernels provide, allows KBMF2K to have better predictive accuracy.

The aim of the PMF method is to decompose the connectivity matrix \( R \) of a bipartite graph with \( N \) drugs and \( M \) targets as a product of two other matrices that express each drug/target in terms of \( D \) latent variables. Matrix \( R \) is defined in the common following way:

\[
R_{i,j} = \begin{cases} 
1 & \text{drug } i \text{ interacts with target } j \\
0 & \text{otherwise}
\end{cases} \tag{2.11}
\]

After that formulation, the main goal is to compute the best possible approximation of the latent variables while avoiding overfitting. The predicted interaction matrix is then computed:

\[
\hat{R}_{N\times M} = U_{N\times D}^T V_{D\times M} \tag{2.12}
\]

Because PMF models interactions by probabilistic linear models with Gaussian noise, the conditional probability over the known interactions in \( R \) is given by:

\[
P(R|U, V, \sigma^2) = \prod_{i=1}^{N} \prod_{j=1}^{M} [f(R_{i,j}|u_i^T v_j, \sigma^2)]^{I_{i,j}} \tag{2.13}
\]

where:

- \( n \): number of drugs
- \( m \): number of targets
- \( f(x|\mu, \sigma^2) \): Gaussianly distributed probabilistic density function for \( x \) with:
  - \( \mu \): mean value
\[ \sigma: \text{variance} \]

\[ I_{i,j}: \text{indicator function equal to 1 if } R_{i,j} \text{ has a known interaction and 0 otherwise} \]

Using zero-mean, spherical Gaussian priors on the latent vectors, we arrive on the following log-likelihood formulation for \( U \) and \( V \):

\[
\ln(p(U, V | R, \sigma^2, \sigma^2_U, \sigma^2_V)) = -\frac{1}{2\sigma^2} \sum_{i=1}^{N} \sum_{j=1}^{M} I_{i,j} (R_{i,j} - u_i^T v_j)^2 \\
- \frac{1}{2\sigma^2_U} \sum_{i=1}^{N} u_i^T u_i \\
- \frac{1}{2\sigma^2_V} \sum_{j=1}^{M} v_j^T v_j \tag{2.14}
\]

In the above formula, the first term on the right-hand side is the squared error function to be minimized and the last two terms are regularization terms that are added to avoid overfitting. The end goal is to find the \( U \) and \( V \) matrices that maximize the log likelihood function.

Zheng et al. (2013) considered a factor model, called Multiple Similarities Collaborative Matrix Factorization (MSCMF) for predicting drug-target interactions. This formulation is based on weighted low-rank approximation for predicting drug-target interactions and uses multiple types of similarity matrices for both drugs and targets, as an improvement on the traditional collaborative matrix factorization method (CMF). As it has been described in the previous methods, the main goal of matrix factorization is to approximate the connectivity matrix \( R \) as a product of two other matrices \( U \) and \( V \) such that:

\[ R \approx U^T V \tag{2.15} \]

Standard CMF introduces regularization terms to ensure that \( UU^T \approx S_d \) and \( VV^T \approx S_t \). It also formulates the following objective function:

\[
\min_{U,V} \| W \cdot (R - UV^T) \|_F^2 \\
+ \lambda_t (\| U \|_F^2 + \| V \|_F^2) \\
+ \lambda_d \| S_d - UU^T \|_F^2 \\
+ \lambda_t \| S_t - VV^T \|_F^2 \tag{2.16}
\]

where:
• $|| \cdot ||_F$: Frobenius norm

• $\lambda_l, \lambda_d, \lambda_t$: parameters

• $W$: weight matrix where $W_{i,j} = 0$ for unknown drug-target pairs

In the above formula, the first term is the weighted low-rank approximation term and the second is the regularization term that helps avoid overfitting. The third and fourth terms are also used for regularization purposes, more specifically to bring similar latent feature vectors of drugs and targets close and latent vectors of dissimilar drugs/targets apart.

The approach proposed by Zheng (MCMF) uses multiple similarities for both drugs and targets. MCMF integrates chemical structure similarity, genomic sequence similarity, ATC similarity, Go similarity and protein-protein network similarity. The objective function of this method is the following:

$$\min_{U,V} ||W \cdot (R - UV^T)||_F^2$$

$$+ \lambda_l(||U||_F^2 + ||V||_F^2)$$

$$+ \lambda_d|| \sum_{k=1}^{M_d} w_d^k S_d^k - UU^T||_F^2$$

$$+ \lambda_t|| \sum_{k=1}^{M_t} w_t^k S_t^k - VV^T||_F^2$$

$$+ \lambda_w(||w_d||_F^2 + ||w_t||_F^2)$$

$$s.t. ||w_d|| = ||w_t|| = 1$$

(2.17)

where:

• $M_d$: number of drug similarity matrices

• $M_t$: number of target similarity matrices

• $\lambda_w$: parameter

• $w_d, w_t$: weight vectors for linearly combining the drug and target similarity matrices.

This strategy generally improves predictive performance but because drugs and targets are projected into low-rank matrices it is expected that a significant amount of information could be lost, leading to a solution that is not optimal.
Ezzat et al. (2017) proposed two matrix factorization methods (GRMF and WGRMF) for drug-target interaction prediction. Both of them implicitly perform manifold learning via graph regularization. The only difference between these two methods is that WGRMF utilizes a weight matrix \( W \) in the same way that CMF does. The weight matrix basically prevents unknown drug-target pairs from contributing to the final predictions. Otherwise, unknown interaction pairs would count as noninteractions (negative instances) and may negatively affect predictions.

The general method consists of two steps:

1. **WKNKN** (weighted k nearest known neighbors). It is a preprocessing step that transforms all the 0’s in the original drug-target interaction matrix into interaction likelihood values. This is the basic differentiating characteristic compared to the other methods that treat all zero entries as noninteraction examples. The researches observed that this step improves the results for all methods except those that utilize a weight matrix. Additionally, when WKNKN is used with the already faster GRMF, the predictive performance becomes comparable with WGRMF.

2. **GRMF** (graph regularized matrix factorization) or WGRMF (graph regularized matrix factorization) for predicting drug-target interactions.

GRMF’s objective function can be written as:

\[
\begin{align*}
\min_{U,V} & \quad \| R - UV^T \|_F^2 \\
& + \lambda_t (\| U \|_F^2 + \| V \|_F^2) \\
& + \lambda_d \text{Tr}(U^T L_d U) \\
& + \lambda_t \text{Tr}(V^T L_t V)
\end{align*}
\]

(2.18)

where:

- \( \text{Tr}(\cdot) \): trace of a matrix
- \( L_d, L_t \): graph laplacians
By adding the weight matrix $M$ to the objective function 2.13 we get the following objective function for WGRMF:

$$
\min_{U, V} ||W \cdot (R - UV^T)||_F^2
+ \lambda_t(||U||_F^2 + ||V||_F^2)
+ \lambda_d Tr(U^T L_d U)
+ \lambda_t Tr(V^T L_t V)
$$

(2.19)

Liao et al. (2016) proposed a drug-target interaction prediction method which utilizes matrix completion based on the singular value thresholding (SVT) algorithm. This method is based only on interaction info and introduces on top of the observed interaction entries (positive instances) that other techniques like (Net-LapRLS) use, negative instances to improve the prediction accuracy.

Fan et al. (2018) proposed the neighborhood constraint matrix factorization (NCMC) approach. In their work, by applying a neighborhood constraint, they were able to use side information to recover a matrix with fewer known entries and also predict interactions for new drugs and targets without any observed interactions.

Yu et al. (2018) introduced a drug-drug interaction (DDI) prediction method which utilizes non-negative and semi-nonnegative matrix factorization. Their approach could predict not only conventional binary DDIs but also whether the interaction increases or decreases the behavior of the interacting drugs before someone made a co-prescription.

Luo et al. (2018) proposed a drug repositioning recommendation system (DRRS) that they applied on a heterogeneous network that integrated individual networks based in drug similarities, disease similarities, and known drug-disease interactions. The recommendation system they designed is based in the SVT algorithm as well as the faster and more efficient recycling rank-revealing randomized singular value decomposition algorithm (R4SVD).

Liu et al. (2016) introduced the neighborhood regularized logistic matrix factorization (NRLMF) method to predict potential drug-target interactions. This approach models the interaction probability for every drug-target combination using matrix factorization and by assigning higher importance to the know and verified drug-target pairs. In addition, they improve the prediction accuracy by considering the influence of the neighborhood that is created by the most similar drug and targets.
Chapter 3

Our approach

3.1 Data

We used two datasets that have some differences in the type of information that they provide. The first dataset is the one that is considered as the gold standard in the area of drug-target interaction prediction. It was firstly introduced by Yamaniishi et al. (2008). The data set is publicly available and contains four different types of drug-target interactions networks. Each network contains a different type of protein targets, namely, enzymes, ion-channels, G-protein-coupled receptors, and nuclear receptors. All this drug target interaction information was extracted from the following databases: DrugBank (Wishart et al., 2007), KEGG BRITE (Kanehisa et al., 2007), BRENDA (Schomburg et al., 2004) and SuperTarget (Günther et al., 2007).

The second dataset comes from Unterthiner et al. (2014). They used a publicly available drug activity database, called ChEMBL. Its size resembles in-house databases of Big Pharma companies, something that the gold-standard dataset is too small to achieve. This dataset contains more than 1,200 targets, 1.3 million compounds and 13 million ECFP12 features.

http://web.kuicr.kyoto-u.ac.jp/supp/yoshi/drugtarget/
https://www.ebi.ac.uk/chembl/
CHAPTER 3. OUR APPROACH

Table 3.1: This table contains information about the gold standard and ChEMBL datasets. More specifically it contains the number of compounds, protein targets and positive interactions as well as the average, minimum and maximum imbalance ratio (ImR). Imbalance ratio of a label is the ratio of the number of examples that belong to the majority class over the number of examples that belong to the minority class.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Compounds</th>
<th>Proteins</th>
<th>Interactions</th>
<th>Mean(ImR)</th>
<th>min(ImR)</th>
<th>max(ImR)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enzyme</td>
<td>418</td>
<td>664</td>
<td>2926</td>
<td>251.67</td>
<td>417</td>
<td>6.08</td>
</tr>
<tr>
<td>Ion Channel</td>
<td>191</td>
<td>204</td>
<td>1476</td>
<td>63.08</td>
<td>190</td>
<td>4.61</td>
</tr>
<tr>
<td>GPCR</td>
<td>203</td>
<td>95</td>
<td>635</td>
<td>94.27</td>
<td>202</td>
<td>5.58</td>
</tr>
<tr>
<td>Nuclear Receptor</td>
<td>54</td>
<td>26</td>
<td>90</td>
<td>26.97</td>
<td>53</td>
<td>2.37</td>
</tr>
<tr>
<td>ChEMBL</td>
<td>743,339</td>
<td>5,069</td>
<td>2,103,018</td>
<td>8191.51</td>
<td>20.29</td>
<td>43943.06</td>
</tr>
</tbody>
</table>

3.1.1 The Pre-processing on gold standard datasets

The number of proteins in these 4 different datasets are 664, 204, 95 and 26. These proteins interact with, respectively, 445, 210, 223 and 54 drugs through 2926, 1476, 635 and 90 positive interactions. A brief description of the datasets is given in Table 1. This final dataset is produced after a crawling script is run. This script has as input the interaction datasets that we obtain from http://web.kuicr.kyoto-u.ac.jp/supp/yoshi/drugtarget/. For every compound id we first obtain the ChEMBL code from https://www.kegg.jp/, and then we capture the smiles representation from https://www.ebi.ac.uk/chembl/. The result of these two steps is that we have the smiles representation of every compound in the dataset. This smiles representation is then given to the RDKit library https://www.rdkit.org/ so that we can obtain the extended connectivity fingerprints or ECFPs that will be used as features in the experiments.
CHAPTER 3. OUR APPROACH

Figure 3.1: Example of a chemical compound and its smiles representation

Extended-Connectivity Fingerprints

Extended connectivity fingerprints (ECFPs) provide high dimensional binary representations that are regarded as the best performing drug description in drug discovery applications (Rogers and Hahn, 2010). ECFPs are circular topological fingerprints initially designed to capture molecular features relevant to molecular activity. Despite not being designed for substructure searching but they are well suited for predicting drug activity. They can also be used similarly to other molecular fingerprints in methods like similarity searching, clustering, and virtual screening.

The following are the main properties of ECFPs:

- They use circular atom neighborhoods to represent the structure of a molecule.
- They can be generated quickly using an easily understood method.
- Because they are not defined a priori, they can represent novel structural classes.
- The features are defined to contain both positive and negative structural information (that is, both what is and what is not present), something crucial for analyzing molecular activity.
- They have a customizable method of generation that can produce various types of circular fingerprints for diverse applications.

ECFPs are defined by the three following parameters:
1. Diameter: This parameter defines the maximum diameter for each atom’s circular neighborhood. This parameter in terms controls the number and maximum size of the neighborhood. ECFPs are also identified by this parameter (ECFP4 when the maximum diameter is 4 of ECFP12 when the maximum diameter is 12).

2. Length: This parameter defines the length of the bit string representation. The default length in most libraries is 1014 but larger lengths decrease the probability of bit collisions. The increase of length decreases information loss but at the same time, a machine learning method requires more computation time and storage to handle it.

At this point, it should be noted that the number of compounds for each protein type present in table 3.1 may be different than the number state above. This can happen because some compound don’t have the required information in some of the websites that we collect the data. For example, the kegg website doesn’t contain the link to the ChEMBL site for some compound ids, from where we collect the smiles representation. Similarly, the ChEMBL website may be missing the smiles representation for some chembl id, which then stops us from retrieving the ECFP4 signature from the RDKit library.
3.1.2 The Pre-processing on the ChEMBL dataset

After the researchers applied thresholds to the activity data that they extracted from the ChEMBL database the final dataset contained 2,103,018 activity measurements across 5,069 protein targets and 743,339 chemical compounds. Because it is very important that compounds that share a scaffold, do not coexist in different folds, the researchers also clustered the compounds using single linkage clustering to guarantee a minimal distance between train and test set. This method produced 400,000 clusters that were partitioned into three folds of roughly equal size.

The number of interactions of each protein target varied significantly. Some targets have over 50,000 interactions while others have very few measurements. Furthermore, there is great label imbalance for many of the targets in the dataset.
This bias stems from the fact that researchers are more likely to experiment with compounds that have the potential to return be highly active with protein targets. To ensure that each target has sufficient samples for our models to train on, we used only the targets that have more than 15 active instances and more than 15 inactive instances. The enforcement of this limit to out data left us with 1,230 targets.

The result of all this pre-processing was the production of mainly four files, which we used as the input of our implementation:

- **chembl.fpf:** this file contains 1,318,187 records. Each line begins with the id of a chemical compound (e.g., CHEMBL153534) and continues with the features. Each feature consists of the id, the character ':' and the value 1. The features are represented as ECFP12 fingerprints. This is in essence a sparse representation where the ones represent substructures that the compound has.

  ![Figure 3.4: sample from the chembl.fpf file](image)

- **SampleIdTable.txt:** This file contains 1,318,187 records. Each line contains only the id of a chemical compound which is in the same format with the id in the chembl.fpf file.

- **cluster.info:** this file contains 1,318,187 records. Each line has two columns. The first column has the id of the assigned cluster and the second column has the id of the chemical compound (there are three folds-ids and the possible values are 0,1,2). In this file the id of the chemical compound does not have the same formatting as the corresponding ids in the chembl.fpf and SampleIdTable.txt files. On this file, the value of the new id corresponds to the location of the old id in the file SampleIdTable.txt.

- **targetActivities.txt:** This file contains 3,172,523 records and each line has three columns. The first column contains the id of the type of activity that
is observed for a compound-target pair. The possible values, as well as their meaning, are the following:

- 1: inactive
- 11: weak inactive
- 3: active
- 13: weak active
- 2: unknown
- 0: contradicting
- 10: weakly contradicting

The second column has the compound’s id (its format is the same as the id in the cluster.info file) and the third column has the biological target’s id.

We also had to implement additional preprocessing steps to be able to use the interaction information in the above files.
In the first step of the implementation, the id of every compound in the chembl.fpf file is replaced based on its location in the SampleIdTable.txt file. For example if the compound with id CHEMBL153534 is located in the tenth line of the SampleIdTable.txt, then the new id of the compound will be 10. The above step is implemented so that we can utilize the information located in the cluster.info and TargetActivities.txt files. The next step involves the use of the cluster.info file and the goal is to use its cluster assignments to split the chembl.fpf file in train and test sets. This process is repeated three times so that in every iteration two folds produce the train set and the third fold produces the test set.

We then use the training set to map the train and test set features to a new space where the ids range starts from one and ends in the total number of features. At this stage, we also discard features with frequencies less than a desired threshold (our threshold was 100 compounds per feature). Features that were found only in the test set and not in the train set were also rejected, as the trained models cannot recognize them and therefore use them in their predictions.

In the next step we use the targetActivities.txt file’s information so that for each biological target, only the active and inactive chemical compounds are kept. At this point, it is important to note an additional step that filters the targets to address the unbalanced nature of the data. Targets that contain fewer than 15 active and 15 inactive samples are discarded. With this step, we try to ensure that every trained model will not show bias in predictions towards the majority class.
Figure 3.8: Number of compounds that were measured per target, i.e., the amount of available data for each single target-prediction task.
3.2 Machine learning methods

The first method that we used on our imbalanced dataset was proposed by Liu and Tsoumakas (2018). The main method is called Ensemble of Classifier Chains with Random Undersampling (ECCRU) and it has two variants, ECCRU2 and ECCRU3. All three versions use as a starting point an Ensemble of Classifier Chains (ECC). ECC is a quite popular multi-label learning algorithm that has shown state-of-the-art predictive performance. The main advantage, when compared to simpler multi-label methods like Binary Relevance (BR), lies in the ability of classifier chains to exploit high-order label correlations.

To further explain the methods, we have to define the following notation:

- \( X = \mathbb{R}^d \): \( d \)-dimensional feature space
- \( L = \{l_1, l_2, ..., l_q\} \): label set of \( q \) labels
- \( Y = \{0, 1\}^q \): \( q \)-dimensional label space
- \( D = \{(x_i, y_i)|1 \leq i \leq n\} \): training set with \( n \) instances

Figure 3.9: Number of measurements that were taken for each compound, i.e., to how many different tasks each single compound contributes
• \((x_i, y_i)\): instance that consists of a feature vector \(x_i \in X\) and label vector \(y_i \in Y\).

• \(y_{ij}\): \(j\)-th element of \(y_i\)

\[
Y_{ij} = \begin{cases} 
1 & \text{if } l_j \text{ is associated with } i\text{-th instance} \\
0 & \text{if } l_j \text{ is not associated with } i\text{-th instance} 
\end{cases} 
\] (3.1)

• \(m_j = \min(|D^0_j|, |D^1_j|)\): number of minority class examples.

• \(M_j = \max(|D^0_j|, |D^1_j|)\): number of majority class examples.

• \(ImR_j = M_j/m_j\): imbalance ratio of label \(j\).

• \(h : X \rightarrow \{0, 1\}^q\): the mapping function that the multi-label method learn from \(D\).

### 3.2.1 Classifier Chain Model (CC)

The Classifier Chain model chains a random sequence \(CH\) of \(|L|\) binary classifiers (\(CH_j\) is the index of the label in \(L\)). The \(j\)-th classifier \(h_j\) in the chain, deals with a binary classification problem associated with a label \(l_j\). The characteristic that gives this method the ability to exploit high-order relations is that the feature space of each link in the chain is extended with the values of the previous labels in the chain. Each classifier \(h_j\) in the chain is responsible for learning and predicting the values for \(l_j\), given a feature space that is augmented by all previous predictions in the chain. This flow of information between the binary classifiers, allows this method to take into account possible label correlations but at the same time prohibits any parallel implementations.

### 3.2.2 Ensembles of Classifier Chains (ECC)

Because the performance of CC is clearly affected by the order of the chain, an ensemble framework with different random chain orders is usually used. The ECC algorithm trains \(c\) classifier chains and each CC is trained with a random sequence of labels as well as a random subset of \(D\). Each different training set \(D\) is constructed by sampling with replacement the initial training set \(D(|D| = |D'|)\).

Each trained classifier chain model is likely to be diverse and provide different predictions. The prediction of the ensemble for an unseen instance is obtained by the following generic voting scheme:
Each $k$-th model predicts a vector $y_k = (l_1, l_2, ..., l_{|L|}) \in \{0, 1\}^{|L|}$. The sums of all the predictions are stored in a vector $W = (\lambda_1, \lambda_2, ..., \lambda_{|L|}) \in \mathbb{R}^{|L|}$. From $W$ we can calculate the sum of votes for a label $l_j \in L$ with the following sum:

$$\lambda_j = \sum_{k=1}^{c} l_j \in y_k$$  \hspace{1cm} (3.2)

We can then normalize $W$ to calculate the distribution of scores for each $l_j \in [0, 1]$ and then use a threshold to choose the final prediction $Y$ such that $l_j \in Y$ ($\lambda_j \geq t$) for some threshold $t$.

### 3.2.3 Ensembles of Classifier Chains with Random Undersampling (ECCRU)

The main purpose of the ECCRU method is to overcome the limitations that classic multi-label classification methods face, when they are applied to highly imbalanced datasets. This is done by combining the classifier chain method with random undersampling (Breiman, 2017) to balance the class bias of each label. The pseudocode, as shown by (Liu and Tsoumakas, 2018) is presented in Algorithm 1.

**Algorithm 1** Training of CCRU

**Input:** multi-label data set: $D$, sequence of labels: $CH$

**Output:** CCRU model: $h = \{h_{1}, \ldots, h_{|CH|}\}$

1: $D_1 \leftarrow \{(x_1, y_{CH,1}), \ldots, (x_{|D|}, y_{|D|CH})\}$

2: $h \leftarrow \emptyset$

3: for $j \leftarrow 1$ to $|CH|$ do

4: $D_j^* \leftarrow \text{RandomUnderSample}(D_j)$

5: train $h_j$ based on $D_j^*$

6: $h \leftarrow h \cup h_j$

7: if $j < |CH|$ then

8: $D_{j+1} \leftarrow \emptyset$

9: for $(x, y)$ in $D_j$ do

10: $\hat{y}_{CH,j} \leftarrow h_j(x)$

11: $x' \leftarrow [x_1, \ldots, x_d, \hat{y}_{CH,1}, \ldots, \hat{y}_{CH,j}]$

12: $D_{j+1} \leftarrow D_{j+1} \cup (x', \hat{y}_{CH,j+1})$

end for

end if

end for

16: return $h = \{h_{1}, \ldots, h_{|CH|}\}$
The basic version of the ECCRU method builds an ensemble of several chains in the same way that the ECC classifier does. The two algorithms combine multiple CCRUs or CCs that have been generated from different label sequences and training subsets. The main difference is that before the training of each binary classifier, random undersampling of its majority class examples is performed. By randomly selecting $m_j$ examples from label’s $l_j$ majority class we end up with a perfectly balanced dataset that can be used for training.

Another difference, when compared to the classic ECC algorithm, is that the ECCRU method does not use the true label’s values to augment the feature space but instead uses the in-sample and out of sample estimates of the values of these labels. These estimates are actually the predictions of the corresponding binary models on the entire training set.

This implementation greatly decreases the probability of a majority class example being used in a chain. A simple way to overcome this issue is to increase the number of chains. However, this approach also increases the computational cost. The ECCRU2 and ECCRU3 methods where introduced as a solution to this problem. Their proposal is to redistribute the computational cost by building a different number of binary classifiers per label. More specifically, this number has to be inversely proportional to its number of minority examples. We present the example that was given in (Liu and Tsoumakas, 2018) to better understand the proposed variations.

If we consider a dataset with 100 training examples and 3 labels, each with 10, 20 and 30 minority examples, each classifier will be trained on 20, 40 and 60 examples respectively. In an ensemble with 10 chains, the total number of used majority examples will be $10 \times (10 + 20 + 30)$. The ECCRU2 and ECCRU3 approaches divide the computational cost equally, so each label will have 200 majority examples. The next step is to calculate the number of binary classifiers $c_j$ that will have to be built for every label with the following formula:

$$c_j = \left\lfloor \frac{c \sum_{k=1}^{q} m_k}{q \cdot m_j} \right\rfloor$$

(3.3)

Where $c$ is the number of chains and $q$ the number of labels.

Because the $c_j$ value for each label is different, ECCRU creates chains of different lengths. To continue the aforementioned example the ECCRU2 algorithm would create 6 chains with all three labels and 4 chains that only include the first two labels. After that point, the creation of CCs with less than two labels is meaningless. The pseudocode for ECCRU2, as shown in (Liu and Tsoumakas, 2018) is presented below. To limit the number of classifiers when highly imbalanced
dataset are used, the algorithm sets an upper limit to the allowed number of classifiers. This limit is the value $c_{\text{max}}$ and is defined as a multiple of the total number of chains $c$: $c_{\text{max}} = c \theta_{\text{max}}$. The proposed values for $c$ and $\theta_{\text{max}}$ both are both equal to 10, and so the $c_{\text{max}}$ is 100. The only change in ECCRU3 is the addition of a lower limit $c_{\text{min}}$ which is a multiple of the total number of chains $c$: $c_{\text{min}} = c \theta_{\text{min}}$. $\theta_{\text{min}}$ is also bounded in $\frac{1}{c} \leq \theta_{\text{min}} \leq 1$ to ensure that $1 \leq c_{\text{min}} \leq c$. In this variant the final $c_j$ value is computed as $\min\{\max\{c_j, c_{\text{min}}\}, c_{\text{max}}\}$. The proposed values for $c$ and $\theta_{\text{min}}$ are 10 and 0.5 respectively, so $c_{\text{min}} = 5$.

Algorithm 2 Training of ECCRU2

**Input:** multi-label data set: $D$, number of labels: $q$, standard number of chains: $c$, the coefficient of maximal number of chains: $\theta_{\text{max}}$

**Output:** ECCRU model: $h = \{h_1, \ldots, h_{c'}\}$

1: for $j \leftarrow 1$ to $q$ do
2: calculate $c_j$ according to (3.3)
3: $c_j \leftarrow \min\{c_j, c \theta_{\text{max}}\}$
4: $c_n_j \leftarrow c_j$
5: end for
6: $c' \leftarrow 0$
7: for $i \leftarrow 1$ to $\theta_{\text{max}}$ do
8: $S \leftarrow \emptyset$
9: for $j \leftarrow 1$ to $q$ do
10: if $c_n_j > 0$ then
11: $S \leftarrow S \cup j$
12: $c_n_j \leftarrow c_n_j - 1$
13: end if
14: end for
15: if $|S| < 2$ then
16: break
17: end if
18: $CH^i \leftarrow \text{RandomPermute}(S)$
19: $D' \leftarrow \text{SampleWithReplacement}(D)$
20: $h_i \leftarrow \text{TrainCCRU}(D', CH^i)$
21: $c' \leftarrow c' + 1$
22: end for
23: $h \leftarrow \{h_1, \ldots, h_{|CH|}\}$
24: return $h = \{h_1, \ldots, h_{|CH|}\}$

In the ECCRU2 algorithm the number of binary classifiers contained in a CCRU $h^i$ does not always equal $q$. So, the classic regularization of the output
using the number of classifiers would be wrong. For that reason a \( q \) dimensional vector \( cc \) is introduced to count the number of classifiers used for each label. This vector is used to normalize the output vector as shown in line 14 of Algorithm 3.

**Algorithm 3** Testing of ECCRU2

**Input:** test instance \( x \), number of labels: \( q \), ECCRU2 model: \( h = \{ h^1, h^2, \ldots, h^c' \} \)

**Output:** prediction label vector \( \hat{y} \)

1: \( \hat{y} \leftarrow 0 \)
2: \( cc \leftarrow 0 \)
3: **for** \( i \leftarrow 1 \) to \( c' \) **do**
4: **for** \( j \leftarrow 1 \) to \( |h^i| \) **do**
5: \( k \leftarrow \) the index of label trained by \( h^i_j \)
6: \( cc_k \leftarrow cc_k + 1 \)
7: \( x' = [x_1, \ldots, x_d, h^i_1(x), \ldots, h^i_{j-1}(x)] \)
8: **if** \( h^i_j(x') = 1 \) **then**
9: \( \hat{y}_k \leftarrow \hat{y}_k + 1 \)
10: **end if**
11: **end for**
12: **end for**
13: **for** \( j \leftarrow 1 \) to \( q \) **do**
14: \( \tilde{y}_j \leftarrow \hat{y}_j / cc_j \)
15: **end for**

### 3.2.4 Multi-Label Co-Training (MLCT)

The majority of multi-label learning methods use only labeled examples in their training. In cases where the number of labels is substantial, the exponential size of their powerset, requires a very large number of labeled examples. The reality is that in many domains, the collection of a sufficient number of examples is very costly or sometimes even impractical. Most multi-label methods that try to leverage both labeled and unlabeled examples are typically transductive and thereby cannot make predictions for unseen examples. Co-training is an important branch of semi-supervised learning, that can utilize labeled as well as unlabeled samples and has a natural inductive classification ability. This means that this type of classifier can be used for the prediction of unseen examples. \cite{Xing2018} proposed a Multi-Label Co-Training method that can perform inductive multi-label classification and utilize labeled and unlabeled examples. This approach had to overcome the two following issues:

- How to solve the class-imbalance problem that characterizes the majority of
multi-label datasets. Typically, the number of relevant samples can vary significantly across labels and it is much smaller than the number of irrelevant samples. This general problem can be exaggerated in the label communication stage of the co-training algorithm.

- How to select samples with confidence, and communicate their predicted labels among classifier when these samples can be associated with several labels and not just one.

The basic workflow in an iteration of the MLCT algorithm is the following:

1. Independently train models on different views of the dataset and make predictions on unlabeled examples.
2. Adjust the predicted likelihoods using the co-occurrence information from the labels.
3. Summarize the adjusted likelihoods across views and calculate the confidence of every sample based on these summarized likelihoods.
4. Select a number of samples with the highest confidence values.
5. Apply label-wise filtering on the likelihoods of the select samples.
6. Communicate the filtered labels among models.

The prediction on unseen samples is performed via a majority vote of every classifier.

The original co-training method ([Blum and Mitchell, 1998b]) uses the notion of views of the feature space. We have an instance space $X = X_1 \times X_2$, where $X_1$ and $X_2$ correspond to two different views of an example. That is, each example $x$ is given as a pair $(x_1, x_2)$. We assume that each view in itself can provide sufficient and independent information to produce a classifier with good generalization. In our implementation, we split features in half to create to equal sized feature views. The following are some definitions that will help to explain in more detail the MLCT algorithm:

- $X = \{X^v\}_{v=1}^m$: m view representations of n samples where $X^v \in \mathbb{R}^{n \times d_v}$
- $x_j^v \in \mathbb{R}^{1 \times d_v}$: $d_v$-dimensional feature vector for the $j$-th sample of the $v$-th view.
- $y_j \in \{0, 1\}^q$: $q$-dimensional feature vector for the $j$-th sample
• $\mathcal{Y}_{j,c} = \begin{cases} 
1 & c\text{-th label is relevant for the sample} \\
0 & c\text{-th label is irrelevant for the sample} 
\end{cases} \quad (3.4)$

where $1 \leq c \leq q$

• $L = \{(x_j, y_{j})\}_{j=1}^{l}$: labeled set with $l$ samples

• $U = \{(x_j)\}_{j=l+1}^{n}$: unlabeled set with $u = n - l$ samples.

The MLCT algorithm can be divided in the following two steps:

1. **Label Correlation**

The MLCT algorithm uses label correlations to address the class-imbalance problem. Positive correlation between two labels $c_1$ and $c_2$ means they often co-occur. This measure can be calculated in the following way:

$$C(c_2, c_1) = \frac{\sum_{j=1}^{l} [y_{j,c_1} = 1][y_{j,c_2} = 1]}{\sum_{j=1}^{l} [y_{j,c_1} = 1]} \quad (3.5)$$

$C(c_2, c_1)$ represents the probability that an example is labeled as $c_2$, given that it is already labeled as $c_1$. Because of the formulation of the correlation metric, $C(c_2, c_1) \neq C(c_1, c_2)$.

Assuming that $f_{v}^{x_j} = [f_{v,1}^{x_j}, \ldots, f_{v,q}^{x_j}] \in \mathbb{R}^q$ is the likelihood of $x_{j}^{v}$ to belong to $q$ labels in the $v$-th view. To overcome the class imbalance problem, MLCT predicts confidence values for every label using label correlations as follows:

$$p_{j,c}^{v} = \frac{1}{1 + e^{-(w_{j,c}^{+} - w_{j,c}^{-})}} \in (0, 1) \quad (3.6)$$

where $p_{j,c}^{v}$ is the updated likelihood of the $c$-th label for $x_j^v$, $w_{j,c}^{+}(w_{j,c}^{-})$ represent the confidence that the $c$-th label is relevant( irrelevant) to $x_j^v$ and are calculated as follows:

$$w_{j,c}^{-} = 1 - w_{j,c}^{+} \quad (3.7)$$
\[ w_{j,c}^+ = f_{j,c}^v \frac{\sum_{c=1}^{q} f_{j,c,c}^v C(c, c2) \sum_{c=2=1} \Delta(C(c, c2))}{\sum_{c=1}^{q} \Delta(C(c, c2))} - 1 \]  
(3.8)

where \( \Delta(a) = 1 \) when \( a > 0 \), and 0 otherwise. \( \sum_{c=2=1}^{q} f_{j,c,c}^v C(c, c2) \) measures how to correlation between the \( c \)-th label and all the other labels, contributes to the relevance that the \( c \)-th label has with \( x_j^v \). In addition, \( \sum_{c=1}^{q} \Delta(C(c, c2)) - 1 \) counts how many labels are positively correlated with the \( c \)-th label.

2. Communication of Label Information

This step involves the selection of the most confident samples, and communication of their predicted labels among classifiers. The traditional co-training algorithm selects the most confident samples for communication. However, the multi-label environment proves that this task is far more complicated because most samples are usually relevant with multiple labels.

The overall prediction reliability of the \( j \)-th sample for the \( c \)-th label is calculated as follows:

\[ \tilde{h}_{j,c}^v = \frac{1}{1 - m} \sum_{v=1}^{m} \sum_{v'=1,v'=v}^{m} |p_{j,c}^v - (1 - p_{j,c}^v')| \]  
(3.9)

where a higher \( \tilde{h}_{j,c}^v \) value indicates a higher agreement across \( m-1 \) classifiers about the relevance between the \( c \)-th label and the \( j \)-th sample. The overall prediction reliability of the \( j \)-th sample can be calculated by summing up over \( q \) labels in the following way:

\[ \tilde{\gamma}_j^v = \frac{1}{q} \sum_{c=1}^{q} \tilde{h}_{j,c}^v \]  
(3.10)

where, like the previous equation, larger \( \tilde{\gamma}_j^v \) equal a greater consistency across classifiers. This consistency is the main factor that positions the \( j \)-th sample as a good candidate for communication. Based on this notion, the MLCT algorithm then selects the \( u_b \) samples with the largest \( \tilde{\gamma}_j^v \) values to form the
candidate sample set $B_v$. The next step involves the definition of two threshold values per label to identify confident labels amongst the selected examples.

$$
\theta^v_+(c) = \frac{\sum_{x^v_j \in B_v} f^v_{j,c} \cdot [f^v_{j,c} \geq 0.5]}{\sum_{x^v_j \in B_v} f^v_{j,c} \cdot [f^v_{j,c} \geq 0.5]}
$$

(3.11)

$$
\theta^v_-(c) = \frac{\sum_{x^v_j \in B_v} f^v_{j,c} \cdot [f^v_{j,c} < 0.5]}{\sum_{x^v_j \in B_v} f^v_{j,c} \cdot [f^v_{j,c} < 0.5]}
$$

(3.12)

where $\theta^v_+(c)(\theta^v_-(c))$ is the average predicted likelihood of the $c$-th label on the $v$-th view which is estimated using plausible relevant(irrelevant) samples. The final step before the communication of the examples to the classifiers, is the label-wise filtering using the calculated thresholds in the following way:

$$
b^v_{j,c} = \begin{cases} 
1, & \text{if } f^v_{j,c} > \theta^v_+(c) \\
0, & \text{if } f^v_{j,c} \leq \theta^v_-(c)
\end{cases}
$$

(3.13)

These new calculated labels and the feature representations from the samples in the $v$-th view are used to augment the labeled training set. The pseudocode that was introduced in (Xing et al., 2018) is presented below:
Algorithm 4 MLCT pseudo-code

Input:
L: labeled samples set in m views;
U: unlabeled samples set in m views;
B: buffered unlabeled samples;
t: maximum number of iterations for co-training;
u_a: buffer size;
u_b: number of samples to be communicated;

Output:

\( H^v \): the prediction model on the \( v \)-th \( (1 \leq v \leq m) \) view.

1: for \( iter = 1 : t \) do
2: Estimate label correlation \( C(c2, c1) \) \( (1 \leq c1, c2 \leq q) \) via Eq. (4.13)
3: Randomly pick \( u_a \) samples from \( U \) and put them into \( B \)
4: for \( v = 1 : m \) do
5: Update classifier \( H_v \) based on \( L \) and make predictions on \( B \)
6: Adjust the initially predicted likelihoods \( f_{j,c}(x_j^v \in B) \) via Eq. (4.14)
7: end for
8: for \( v = 1 : m \) do
9: Calculate \( \tilde{r}_j^v \) \( (1 \leq j \leq u_a) \) via Eq. (4.18), then select \( u_b \) samples from \( B \)
10: Compute \( \theta^v_+(c), \theta^v_-(c) \) \( (1 \leq c \leq q) \) via Eq. (4.19,4.20)
11: Apply label-wise filtering on \( f_{j,c}(x_j^v \in B_v) \) via Eq. (4.21), and form the communication label set \( \Delta^v = \{ b_{j,c}^v \} \).
12: end for
13: Communicate \( \Delta = \{ \Delta^v \}_{v=1}^m \) to \( \{ H^v \}_{v=1}^m \), augment the labeled training set \( L = L \cup \Delta \) and reduce the unlabeled training set \( U = U - \Delta \)
14: end for
15: return \( H_{v=1}^{vm} \)
Chapter 4

Results

4.1 Performance Metrics

The datasets that are used in the literature of drug-target interaction prediction are usually highly imbalanced. The number of inactive pairs (negative samples) significantly exceeds that of the active pairs (positive samples). In this type of datasets traditional metrics like accuracy convey a false picture of performance. For example, when a target has a minimal number of positive samples, and a large number of negative samples, we can implement a dummy classifier that only predicts the negative class. In this case, the classifier will achieve a high accuracy despite misclassifying every positive sample, something that is considered inadequate as an evaluation metric.

Two measures that are far more suitable for imbalanced datasets are the area under the Receiver Operating Characteristic (ROC) curve, auROC, and the area under the precision-recall (PR) curve, auPR. These two measures are widely used in the literature of drug-target interaction prediction (Mousavian et al., 2016; Chen and Zhang, 2013; Cao et al., 2012; Chan et al., 2016) and thus have become standard metrics for comparison. To understand how to calculate these two measures we first have to calculate simpler statistical measures using the confusion matrix displayed in figure 4.1.

We first have to make some definitions. Let $TP$, $FP$, $TN$ and $FN$ refer to true positives, false positives, true negatives and false negatives respectively. True positives are correctly classified positive samples by the classifier, and false positives are negative samples falsely classified as positive by the classifier. Conversely, true negatives are negative samples correctly predicted as negatives, and false negatives are positive samples incorrectly predicted as negatives by the classifier.

Sensitivity or true positive rate (TPR) or recall is the proportion of actual pos-
Figure 4.1: Confusion matrix.

Positive samples that are correctly identified as such (i.e., the percentage of active drug-target interaction pairs that are correctly identified as active interactions).

\[ \text{Sensitivity} = TPR = \text{Recall} = \frac{TP}{TP + FN} \]  \hspace{1cm} (4.1)

Precision or positive predictive rate is the proportion of positive predictions that are accurate.

\[ \text{Precision} = \frac{TP}{TP + FP} \]  \hspace{1cm} (4.2)

Specificity or positive predictive rate is the proportion of actual negative samples that are correctly identified as such (e.g., the percentage of inactive drug-target interaction pairs that are correctly identified as inactive interactions).

\[ \text{Specificity} = \frac{TN}{TN + FP} \]  \hspace{1cm} (4.3)

False positive rate (FPR) is the ratio of the number of wrongly classified negative samples to the total number of actual negative samples.

\[ FPR = \frac{FP}{FP + TN} = 1 - \text{Specificity} \]  \hspace{1cm} (4.4)
CHAPTER 4. RESULTS

55

The ROC curve can be obtained by plotting the false positive rate on the x-axis and the true positive rate on the y-axis. For a given classifier, we obtain different (FPR, TPR) pairs at different threshold values. These pairs correspond to points in the ROC space. An ideal classifier has an auROC value of 1 while a random classifier has a value of 0.5. For highly imbalanced datasets, auPR is a better evaluation metric (Davis and Goadrich, 2006). The PR curve can be obtained by plotting the recall on the x-axis and the precision on the y-axis. This score penalizes the false positives more compared to auROC, which makes it more suitable for imbalanced datasets.

4.2 Gold Standard Datasets Results

As we have stated before, we used common evaluation datasets, typically called the gold standard datasets, as a mean to compare prediction performance. These datasets include drug-target interactions for enzymes (E), ion channels (ICs), G protein-coupled receptors (GPCRs) and nuclear receptors (NRs) and were collected from public databases like KEGG, BRITE, BRENDA, SuperTarget, and DrugBank. It was first introduced by Yamanishi et al. (2008) and it is widely used in the area. Although convenient for comparison between different methods in this area, they present serious limitations like the fact that they contain only true positive interactions. This characteristic makes the datasets ignore many important aspects of the drug-target interactions like its quantitative affinity.

Additionally, the majority of the methods developed in the area use prediction formulations that are based on the practically unrealistic assumption that during the construction of the models and the evaluation of their predictive accuracy, we have the full information about the drug and target space. In particular, the typical evaluation method assumes that the drug-target pairs to be predicted in the validation set are randomly distributed in the known drug-target interaction matrix (Figure 4.3). To be able to have comparable results with the state-of-the-art methods shown in Table 4.2 we had to change the typical training and testing process we typically use in our multi-label implementations. Instead of splitting the dataset based on the compounds and therefore falling into the realistic setting described in figure 4.2 we randomly hide drug-target pairs from the train set. In our implementations, the ECCRU variants just ignore these drug-target pairs during the training phase. During testing, the algorithm makes a prediction for every instance and every label in the train set and the accuracy metric e.g. auRoc is computed only based on the hidden values. With this approach the algorithm sees every compound and target during training with the exclusion of some drug-target pairs that are kept for testing.

Our standard multi-label implementation is based on the problem setting shown
CHAPTER 4. RESULTS

Figure 4.2: The validation setting that we used in our approach. Rows represent the different chemical compounds and columns represent the protein targets-labels. The matrix contains active and inactive drug-target interaction pairs (entries colored blue and light blue respectively). The test set consists of one-fifth of the rows of the drug-target interaction matrix, and each of these entries are used simultaneously as test pairs (5 fold CV in drugs).

Figure 4.3: The validation setting used in the majority of published methods. Rows represent the different chemical compounds and columns represent the protein targets-labels. The matrix contains active and inactive drug-target interaction pairs (entries colored blue and light blue respectively). The matrix entries are usually randomly partitioned into five parts, each of which is removed in turn from the training set (entries colored grey) and used as a test data.

in Figure 4.2 which is far more realistic and similar to what a typical pharmaceutical company may encounter, where only part of the drug or target information is available during the model training phase. For example, a molecular chemistry team could synthesize a new chemical compound that needs to be screened with all the available protein targets. The problem with the setting in Figure 4.3 is that its drug-target interaction prediction model is trained on a specific training set (a set
comprised of compounds and protein targets that were available to the company at a specific point in time). So, it is easy to see that the newly synthesized chemical compounds will not be able to be evaluated by a model based in this problem setting. I contrast, it is obvious that our model was developed with the more setting in mind and therefore is more suitable for use. We implemented two different versions for the two different problem settings. The implementation of the unrealistic setting gave us the ability to compare our methods with the state-of-the-art approaches in the area. In our experiments we tried many different combinations with the following parameters:

- number of CCs: we tested ensembles with 10, 50 and 100 classifier chains. We also experimented with bigger ensembles but the results did not show improvements.

- base classifier and kernel/solver: we tested implementations of SVM, Logistic Regression and Multinomial Naive Bayes provided by the sklearn library. The SVM was tested with the linear and RBF kernels, while Logistic Regression was tested with the lbfgs and liblinear solvers.

- classifiers: we tested the standard ensemble of classifier chains algorithm as well as all the variants of the ECCRU algorithm.

The best performing combinations out of the 60 we tested for every gold standard benchmark dataset in terms of the area under the receiver characteristic curve and the area under the precision recall curve are shown in Table 4.1.

Table 4.1: This table contains the top performing combinations in terms of the number of classifier chains per ensemble, base binary classifier in every chain, kernel or solver the ECCRU variant for every benchmark gold standard dataset. The metrics that are displayed are the area under receiver characteristic curve (auROC) and the area under the precision recall curve (auPR). All the methods are tested in the problem setting shown in figure 4.2.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Number of CCs</th>
<th>Base Classifier</th>
<th>kernel/solver</th>
<th>method</th>
<th>auPR</th>
<th>auROC</th>
</tr>
</thead>
<tbody>
<tr>
<td>enzymes</td>
<td>100</td>
<td>SVC</td>
<td>linear</td>
<td>ECCRU</td>
<td>0.4662</td>
<td>0.8460</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Logistic Regression</td>
<td>liblinear</td>
<td>ECCRU</td>
<td>0.3302</td>
<td>0.7530</td>
</tr>
<tr>
<td>ion channel</td>
<td>100</td>
<td>Logistic Regression</td>
<td>lbfgs</td>
<td>ECCRU</td>
<td>0.4802</td>
<td>0.8305</td>
</tr>
<tr>
<td>GPCR</td>
<td>50</td>
<td>Logistic Regression</td>
<td>lbfgs</td>
<td>ECCRU</td>
<td>0.8061</td>
<td>0.9045</td>
</tr>
<tr>
<td>nuclear receptor</td>
<td>100</td>
<td>Multinomial NB</td>
<td></td>
<td>ECCRU</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

When we compare our results with the state-of-the-art methods on the four gold standard datasets (table 4.2) we see that we rank in the last places in terms of the area under. In the enzymes, ion channels datasets we are ranked 7th out of the
CHAPTER 4. RESULTS

Table 4.2: Comparison of the performance of state-of-the-art methods with our best performing implementations on the four benchmark gold standard datasets in terms of the area under receiver characteristic curve (auROC). All the methods are tested in the problem setting shown in figure 4.13.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>DBSI</th>
<th>KBMF2K</th>
<th>Net3BP</th>
<th>Yamamoudi</th>
<th>Yamamoudi</th>
<th>Wang</th>
<th>Mousavian</th>
<th>iDTI-ESBoost</th>
<th>Pallikkala</th>
<th>Our approach</th>
</tr>
</thead>
<tbody>
<tr>
<td>enzymes</td>
<td>0.8075</td>
<td>0.8251</td>
<td>0.9041</td>
<td>0.8920</td>
<td>0.8860</td>
<td>0.9480</td>
<td>0.9089</td>
<td>0.9060</td>
<td>0.8627</td>
<td></td>
</tr>
<tr>
<td>ion channels</td>
<td>0.8029</td>
<td>0.7990</td>
<td>0.8014</td>
<td>0.8530</td>
<td>0.8120</td>
<td>0.8930</td>
<td>0.8890</td>
<td>0.9309</td>
<td>0.9640</td>
<td>0.8046</td>
</tr>
<tr>
<td>GPCRs</td>
<td>0.8024</td>
<td>0.8240</td>
<td>0.8594</td>
<td>0.8530</td>
<td>0.8240</td>
<td>0.8690</td>
<td>0.9025</td>
<td>0.9610</td>
<td>0.9257</td>
<td></td>
</tr>
<tr>
<td>nuclear receptors</td>
<td>0.7578</td>
<td>0.8240</td>
<td>0.8594</td>
<td>0.8530</td>
<td>0.8240</td>
<td>0.8690</td>
<td>0.9025</td>
<td>0.9610</td>
<td>0.9257</td>
<td></td>
</tr>
</tbody>
</table>

10 classifiers, in the GPCRs we are ranked 6th and in the nuclear receptors dataset we are ranked 2nd.

When we compare the performance of our implementations in the two problem settings(last column in table 4.1 and last row in table 4.2) we can see that the performance in the more realistic setting is worse.

Table 4.3: Comparison of the performance of our approach on the four benchmark gold datasets in terms on the area under receiver characteristic curve (auROC) with another state-of-the-art method in an evaluation setting where the compounds in the test set have not been seen during training(figure 4.2).

<table>
<thead>
<tr>
<th>Predictor</th>
<th>Enzymes</th>
<th>Ion channels</th>
<th>GPCRs</th>
<th>nuclear receptors</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pallikkala</td>
<td>0.8320</td>
<td>0.8020</td>
<td>0.8520</td>
<td>0.8460</td>
</tr>
<tr>
<td><strong>Our approach</strong></td>
<td><strong>0.8460</strong></td>
<td><strong>0.7530</strong></td>
<td><strong>0.8305</strong></td>
<td><strong>0.9045</strong></td>
</tr>
</tbody>
</table>

Table 4.4: Comparison of the performance of our approach on the four benchmark gold datasets in terms on the area under precision recall curve (auPR) with another state-of-the-art method in an evaluation setting where the compounds in the test set have not been seen during training.

<table>
<thead>
<tr>
<th>Predictor</th>
<th>Enzymes</th>
<th>Ion channels</th>
<th>GPCRs</th>
<th>nuclear receptors</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pallikkala</td>
<td>0.3950</td>
<td>0.3650</td>
<td>0.4020</td>
<td>0.5180</td>
</tr>
<tr>
<td><strong>Our approach</strong></td>
<td><strong>0.4660</strong></td>
<td><strong>0.3300</strong></td>
<td><strong>0.4800</strong></td>
<td><strong>0.8061</strong></td>
</tr>
</tbody>
</table>

Pahikkala et al. (2014a) detailed this exact problem, proposed some improved validation methods and shared prediction accuracies in the form of auROC and auPR values for the two settings that we have discussed above (tables 4.3 and 4.4). These prediction results were provided for two learning models, the KronRLS (van Laarhoven et al., 2011) and Random Forest methods. From the comparison, we can see that our approach is competitive. In terms of the auRoc score, we have better performance on the Enzymes and nuclear receptors datasets while in terms of the auPR score we achieve better results on the enzymes, GPCRs, and nuclear receptors datasets.

We also tried to test our multi-label co-training (MLCT) implementation on the gold standard datasets but the results were problematic because of their extremely small size. To compute the performance of MLCT, Xing et al. (2018) proposed
CHAPTER 4. RESULTS

the random partition the samples of each dataset into a training set (70%) and a
testing set (30%). For the training set, they proposed to again randomly select 10%
of the samples as the initial labeled data \((L)\) and the remaining 90% as unlabeled
data \((U)\) for co-training. They also suggested to independently repeat the above
partition ten times, and report the average results and standard deviations. As for
the method’s parameters, the researchers proposed the following:

- number of iterations \(t\) is fixed to 30.
- number of samples \((u_a)\) in the buffer pool \(B\) is fixed to \(\lfloor u/t \rfloor\).
- number of samples \((u_b)\) to be shared during the co-training process is fixed
to \(\lfloor 5\% u_a \rfloor\).

In our implementation we reduced the number of iterations to 10 and performed
5-fold cross-validation. We only measured our performance in the enzymes dataset
because the size of the other three is too small to have reliable cross-validated re-
sults. This typical cross-validation strategy leads to the risk of the multivariate
metrics becoming undefined or having large variance because of the small fold
sizes. The base classifiers used in the experiments were the standard ECCRU
method and its two other variants, ECCUR2 and ECCRU3. All these variants
were initialized with 50 Classifier Chains. As base classifiers of the three vari-
ants we selected the 2 best performing methods (Multinomial Naive Bayes and
Logistic Regression) in experiments presented above (table 4.1). We also tested
the performance of the standard ECCRU variants that were trained on the initial
labeled set \(L\).

From the results in table 4.5 we observe that the prediction performance of the
MLCT algorithm outperforms the standard ECCRU variants in almost every case.
We also observe that when we used the Logistic Regressor as the base classifier,
it outperformed the Multinomial Naive Bayes approach in all three variants of the
ECCRU algorithm. In our results, we observed many fluctuations in the auROC
and auPR metrics during the iterations. Because the features are extremely sparse,
in many tests the train set would have around 500 features. This characteristic
creates a lot of problems with the creation of the feature views that were needed
in the MLCT implementation. Some views could have a set of features that would
severely damage the overall performance of the classifier.

The co-training algorithm works well when each of the two or more views of
features that are used provide sufficient and independent information to produce
a classifier with a good generalization capability. So, in our experiments, the low
number of features in every view violated that assumption. In our future work,
Table 4.5: Performance of the standard ECCRU algorithm and the mlct algorithm on all the variants of ECCRU algorithm with 2 different base classifiers for the enzymes dataset in terms of area under Receiver Operating Characteristic curve (auROC) and area under precision recall curve (auPR).

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Base Classifier</th>
<th>Method</th>
<th>auPR</th>
<th>auROC</th>
</tr>
</thead>
<tbody>
<tr>
<td>enzymes</td>
<td>ECCRU</td>
<td>MultinomialNB</td>
<td>0.1444</td>
<td>0.6465</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Logistic Regression</td>
<td>0.1431</td>
<td>0.6865</td>
</tr>
<tr>
<td></td>
<td>MLCT(ECCRU)</td>
<td>MultinomialNB</td>
<td>0.1514</td>
<td>0.6511</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Logistic Regression</td>
<td>0.1398</td>
<td>0.7278</td>
</tr>
<tr>
<td></td>
<td>ECCRU2</td>
<td>MultinomialNB</td>
<td>0.1200</td>
<td>0.6197</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Logistic Regression</td>
<td>0.1427</td>
<td>0.6410</td>
</tr>
<tr>
<td></td>
<td>MLCT(ECCRU2)</td>
<td>MultinomialNB</td>
<td>0.1133</td>
<td>0.6138</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Logistic Regression</td>
<td>0.1633</td>
<td>0.6582</td>
</tr>
<tr>
<td></td>
<td>ECCRU3</td>
<td>MultinomialNB</td>
<td>0.1291</td>
<td>0.6441</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Logistic Regression</td>
<td>0.1464</td>
<td>0.6553</td>
</tr>
<tr>
<td></td>
<td>MLCT(ECCRU3)</td>
<td>MultinomialNB</td>
<td>0.1276</td>
<td>0.6455</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Logistic Regression</td>
<td>0.1954</td>
<td>0.6949</td>
</tr>
</tbody>
</table>

to mediate this problem, we could use different chemical descriptors as different views (e.g. ECFP12 for the first view and ECFP9 for the second view).

4.3 ChEMBL Dataset results

This dataset has been extensively analyzed in a previous section. In our implementation, we had to randomly sample 100000 instances from the original dataset in order to have more manageable runtimes. We also kept the 15 active instances per label low threshold that Unterthiner et al. (2014) proposed in their implementation. We then added a similar threshold for the features (100 instances per feature) in an attempt to reduce the dimensionality and general runtime of the tests. The performance tests of the three ECCRU variants were implemented using a three fold validation that ensured that every chemical compound was only present in one of the folds. The results are presented below:

From tables 4.6, 4.7 we see that our methods are inferior in term of the au-Roc score. This result can be attributed to the severe sampling we performed to be able to have reasonable runtimes. The 8 methods in table 4.7 are trained on the full dataset while we downsample to 100,000 instances. Our inferior performance can be also associated with the ratio between the number of different chain sequences and the total number of labels. A dataset with close to 1000 labels
CHAPTER 4. RESULTS

Table 4.6: Performance of all the variants of ECCRU algorithms on 3 different base classifiers on sample of the CheMBL dataset in terms of the area under the Receiver Operating Characteristic curve (auROC).

<table>
<thead>
<tr>
<th>Base Classifier</th>
<th>method</th>
<th>auROC</th>
</tr>
</thead>
<tbody>
<tr>
<td>ECCRU</td>
<td>MultinomialNB</td>
<td>0.6643</td>
</tr>
<tr>
<td></td>
<td>SVM</td>
<td>0.6714</td>
</tr>
<tr>
<td></td>
<td>Logistic Regression</td>
<td>0.6633</td>
</tr>
<tr>
<td>ECCRU2</td>
<td>MultinomialNB</td>
<td>0.6488</td>
</tr>
<tr>
<td></td>
<td>SVM</td>
<td>0.6492</td>
</tr>
<tr>
<td></td>
<td>Logistic Regression</td>
<td>0.6380</td>
</tr>
<tr>
<td>ECCRU3</td>
<td>MultinomialNB</td>
<td>0.6531</td>
</tr>
<tr>
<td></td>
<td>SVM</td>
<td>0.6601</td>
</tr>
<tr>
<td></td>
<td>Logistic Regression</td>
<td>0.6531</td>
</tr>
</tbody>
</table>

Table 4.7: Performance of several state-of-the-art methods on the CheMBL dataset in terms of the area under the Receiver Operating Characteristic curve (auROC). The methods include Deep Neural Networks, Support Vector Machines (SVM), Binary Kernel Discrimination (BKD), Logistic Regression, k-nearest neighbour, Parzen-Rosenblatt kernel density estimator, Pipeline Pilot Bayesian Classifier (PNPBC) and Similarity Ensemble Approach (SEA).

<table>
<thead>
<tr>
<th>method</th>
<th>auROC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Deep network</td>
<td>0.830</td>
</tr>
<tr>
<td>SVM</td>
<td>0.816</td>
</tr>
<tr>
<td>BKD</td>
<td>0.803</td>
</tr>
<tr>
<td>Logistic Regression</td>
<td>0.796</td>
</tr>
<tr>
<td>k-NN</td>
<td>0.775</td>
</tr>
<tr>
<td>Pipeline Pilot Bayesian Classifier</td>
<td>0.755</td>
</tr>
<tr>
<td>Parzen-Rosenblatt</td>
<td>0.730</td>
</tr>
<tr>
<td>SEA</td>
<td>0.699</td>
</tr>
</tbody>
</table>

can produce an impractical number of different chain sequences. Our available computational power limited us to around 50 different chain sequences in every experiment. These two factors could significantly increase the chance of a poor chain-ordering or error propagation negatively affecting the overall predictive performance. Additionally, the highly imbalanced nature of the dataset could also be a factor that negatively affected the performance. During the undersampling of the ECCRU variants we observed that the number of instances that each label was trained varied from 30 to tens of thousands.

The methods mentioned above make the assumption that all the unknown drug-target pairs can be considered inactive. However, the reality is that many unknown drug-target pairs could actually be active. The ChemBL dataset provides explicit inactive drug-target pairs, something that we used in our implementations and experiments. The results are presented in the table below:
Table 4.8: Performance of all the variants of ECCRU algorithms on 2 different base classifiers on sample of the CheMBL dataset using explicit inactivity information in terms of the area under the Receiver Operating Characteristic curve (auROC).

<table>
<thead>
<tr>
<th>Base Classifier</th>
<th>method</th>
<th>auROC</th>
</tr>
</thead>
<tbody>
<tr>
<td>ECCRU</td>
<td>MultinomialNB</td>
<td>0.5243</td>
</tr>
<tr>
<td></td>
<td>Logistic Regression</td>
<td>0.5391</td>
</tr>
<tr>
<td>ECCRU2</td>
<td>MultinomialNB</td>
<td>0.5184</td>
</tr>
<tr>
<td></td>
<td>Logistic Regression</td>
<td>0.5202</td>
</tr>
<tr>
<td>ECCRU3</td>
<td>MultinomialNB</td>
<td>0.5215</td>
</tr>
<tr>
<td></td>
<td>Logistic Regression</td>
<td>0.5298</td>
</tr>
</tbody>
</table>

The results presented in table 4.8 show that the use of explicit information of inactivity leads to a much harder problem to solve. The predictive performance in terms of the auROC score is inferior for all the variants of the ECCRU algorithm by about 10%. This significant decrease in performance can be attributed to the problem setting itself. The problem setting where all the unknown drug-target interactions are considered inactive gives every classifier the ability to raise its performance artificially. This happens when a model falsely predicts potentially active pairs as inactive and the performance metric i.e. the auRoc score falsely accepts it as correct.
Chapter 5

Future Work

The obvious next step is to run our implementations on the entire dataset and see if the additional number of samples in the training phase can improve the predictive performance. We could also try larger ensembles of classifier chains in an attempt to discover new correlations between the labels. This direction would require far greater computational power because our implementations used no more than 30 processes at a time. Because of the chain nature of our methods, the speed up can be achieved only in the ensemble level, so in a powerful cluster, we could run hundreds of classifier chains in parallel.

Figure 5.1: The validation setting is considered the most challenging in the drug-target interaction prediction problem because neither the drug nor the target of the test pair has been encountered during model training. The matrix contains active and inactive drug-target interaction pairs (entries colored blue and light blue respectively). The grey regions are used in the test set and the black regions contain drug-target pairs that cannot be used during training.
CHAPTER 5. FUTURE WORK

Realistic validation set selection

Figure 5.2: The validation setting that could be used in future work. Rows represent the different chemical compounds and columns represent the protein targets-labels. The matrix contains active and inactive drug-target interaction pairs (entries colored blue and light blue respectively). The test set consists of one-fifth of the columns of the drug-target interaction matrix, and each of these entries are used simultaneously as test pairs (5 fold CV in target proteins).

In terms of our methodology, we could utilize features about the protein targets that are available in the ChEMBL database. By using these new features we can split the dataset based on the targets and therefore test our methods on the other realistic setting (Setting C from section 2.4 or setting in figure 5.2). We could also run simultaneously classifiers that predict the interaction from the drug and target space and then combine the results in one drug target interaction matrix. With this modification, we can experiment with the most challenging setting (Setting D from section 2.4 or setting in figure 5.1) in the drug-target interaction problem, in which we make predictions for drugs and targets that have not been seen during training.

To tackle the class imbalance that all targets have, we could also experiment with other undersampling techniques instead of the random undersampling that the ECCRU algorithm uses. These techniques can include informed under-sampling (Liu et al., 2009; Mani and Zhang, 2003), Synthetic Minority Over-sampling (SMOTE) (Chawla et al., 2002), Tomek links (Tomek, 1976) and Cluster-based sampling (Jo and Japkowicz, 2004).
Bibliography


Dong-Sheng Cao, Shao Liu, Qing-Song Xu, Hong-Mei Lu, Jian-Hua Huang, Qian-Nan Hu, and Yi-Zeng Liang. Large-scale prediction of drug–target interactions using protein sequences and drug topological structures. *Analytica chimica acta*, 752:1–10, 2012.


LY Han, XH Ma, HH Lin, J Jia, F Zhu, Y Xue, ZR Li, ZW Cao, ZL Ji, and YZ Chen. A support vector machines approach for virtual screening of active compounds of single and multiple mechanisms from large libraries at an improved hit-rate and enrichment factor. *Journal of Molecular Graphics and Modelling*, 26(8):1276–1286, 2008.


