Int. J. Advance Soft Compu. Appl, Vol. 13, No. 2, July 2021
Print ISSN: 2710-1274, Online ISSN: 2074-8523

Cost-Sensitive Deep Learning Models for Drug-Target Interaction Prediction

Nassima Aleb

Jubail University College-Computer Science Department
Kingdom of Saudi Arabia
e mail: alebn@ucj.edu.sa

Abstract

Discovering interactions between drug compounds and proteins is fundamental in drug design. However, Drug target interactions (DTIs) prediction is an exceedingly tedious, and onerous task. Consequently, several computational approaches have been elaborated to improve and accelerate the drug discovery procedure. Nevertheless, these methods suffer from the high imbalance rate of the available datasets since there are considerably more non-interacting compound-protein pairs than interacting pairs. This paper presents a contribution in this regard. We propose a deep-learning framework using convolutional recurrent layers for drug-target interaction prediction. This kind of neural networks combines the advantages of both recurrent and convolutional networks. Then, a cost-sensitive learning is performed to improve the performance of our initial model. The experimental analysis shows that our cost-sensitive models outperform similar methods in terms of area under the curve for Receiver Operating Characteristic (auROC) and area under the Precision Recall curve (auPR) metric.

Keywords: *Deep learning models for drug discovery, Drug-target interaction, Imbalanced datasets, Cost-sensitive learning, Convolutional LSTM networks.*

1 Introduction

Detection of interactions within drugs and targets is a primordial phase in drug design and discovery. Therefore, in the initial phases of drug design, the prediction of Drug-Target interaction (DTI) plays a critical role. Nevertheless, this task was, previously, very costly, time-consuming, and prone to errors, as it was mainly based on biological assays and screening methods [1]. Thus, investigating

computational techniques to reduce both costs and errors was indispensable. The unprecedented interest witnessed in the development of new computational approaches for efficient DTI identification has given rise to various approaches for predicting and analyzing DTIs based on the available interaction data [2,3]. The main methods were, essentially, ligand-based methods and docking simulation methods [4]. However, this class of approaches exhibits poor performance and low effectiveness. Nowadays, a huge amount of drugs and genomes heterogeneous data is produced and structured in various openly accessible databases. This has inspired the research community to explore the development of machine learning (ML) methods for efficient prediction of DTIs. ML methods are data-driven techniques whose success, largely, depends on the training datasets. In this paper, we propose a new DL-based approach using convolutional recurrent networks for drug-target interaction prediction. First, we propose a deep-learning model using convolutional LSTM layers for drug-target interaction prediction. This kind of neural networks has never been used before for DTI prediction. It combines the gating of LSTM with 2D convolutions. Input transformations and recurrent transformations are both convolutions. Convolutional networks are famous for their ability in discovering patterns, and LSTMs excel in learning from sequential data while avoiding the problems of long-term dependency. These characteristics motivated our use of CONVLSTM2D since, in our framework, both drugs and targets are represented by raw sequences that present patterns. Then, cost-sensitive learning is performed to improve the performance of our initial model. We implemented various methods to overcome the imbalance rate of the standard datasets. We assessed our methods by using auROC and auPR metrics, which are the most adequate metrics when the datasets are imbalanced. In this work, we used four balancing methods including Tomek links under-sampling [30], SMOTE over-sampling [31], Threshold moving, and Class-weight. As a result, our methods have demonstrated benefits to overcome the bias issue and provided better prediction performance on the usual drug-target datasets. The rest of this paper is structured as follows: Section 2 introduces some related works, Section 3 describes the materials and methods we used in this paper; Section 4 highlights the experimental results and finally, a conclusion in Section 5 ends this paper.

2 Related Work

In an original work on DTI prediction, Yamanishi et al. [5] presented standard datasets composed of four sets of drug-protein pairs. A substantial number of ML algorithms for DTI prediction, using these datasets, has been developed. This comprises various supervised learning approaches as SVM [6], k-nearest neighbor [7], fuzzy logic [8], and random forest [9]. Several methods based on the similarity network have also been proposed. For instance, KronRLS [10] uses 2-dimensional compound similarity-based representations of the drugs and Smith-Waterman similarity representation of the targets. Lately, another method,

SimBoost [11] was suggested for DTI prediction with a gradient boosting machine. The authors use similarity-based information of DT pairs with features extracted from network-based interactions between the pairs. Other methods using similarity networks and matrix factorization are presented in [12-18]. However, like all ML techniques, these methods require a feature engineering phase. Contrasting ML, deep neural networks can automatically extract important features from the input data, combine and integrate low-level features into high-level features, and capture complicated nonlinear relationships in a dataset [19]. Stimulated by its noteworthy success, deep learning-based techniques are now being investigated in many complex domains, including bioinformatics such as in genomics studies [20,21] and quantitative-structure activity relationship (QSAR) studies in drug discovery [22-24]. The most noticeable benefit of deep learning approaches is their ability to extract automatically latent features of the raw data by non-linear transformations in each layer [19]. However, this advantage makes them more dependent on datasets. Nevertheless, in DTI prediction research, datasets constitute a critical point, since many years, most DTI studies employed the four major datasets by Yamanishi et al. [5] in which DT pairs with no known binding information are treated as negative (not-binding) samples. Designing unknown interactions as negative samples strongly affect methods performance. Recently, few DTI methods using datasets with binding information have been developed. They create binarized datasets by using a threshold for binding scores. Some of these methods employ Deep Neural Networks (DNN) for DTI prediction using various input models for proteins and drugs [25,26]. In [27,28], are introduced two approaches based on stacked auto-encoders and deep-belief networks. DL-based algorithms for DTI prediction, diverge from each other regarding two main aspects. The first is the input data representation, particularly, drug features. Some examples are Simplified Molecular Input Line Entry System: SMILES [29], Ligand Maximum Common Substructure (LMCS) Extended Connectivity Fingerprint (ECFP) or a combination of these features. The second aspect is the architecture of the model that is defined using different neural network (NN) types. Commonly, the prediction of DTI starts with the representation of the input data for the drug and target, then diverse NN types with various structures are applied to learn their features separately. The obtained features are then merged and fed to a feedforward neural network for the prediction task. However, one of the major obstacles in drug-target interaction prediction is due to the imbalanced nature of the available datasets. Negative samples extremely outnumber positive samples. This weakness, if not appropriately handled, can reduce drastically the prediction performance of any approach. It has been shown that imbalance handling is strongly related to cost-sensitive learning. In classification problems, cost-sensitive learning is the process of associating a different cost to each type of classification error. Cost-sensitive learning shows a promising way of modifying the learning in the context of class imbalance. However, we need to distinguish between the theory and the practice of this subject. In theory, when it is possible to have a good estimate of the cost of

each type of classification error (false positive and false negative), using cost-sensitive learning requires the definition of a cost matrix. In that case, it will be judicious to use a metric directly linked to this cost matrix. For example, in the case of credit card fraud detection, the datasets are highly imbalanced. The aim being to, on one hand, minimize the number of fraud as much as possible, and on the other hand, don't be too strict with one's clients. In this regard, it will be possible to define a cost function specially tailored for each specific type of error with the help of an expert (a credit analyst). In practice, it is not always possible, or even suitable, to have an expert to assign error costs. Therefore, it is usual to apply a cost inversely proportional to the class imbalance. Furthermore, it has been indicated that learning when costs are different and unknown, and learning from imbalanced datasets can be handled in a similar way [29]. Indeed, the challenge in classification is to find an adequate decision boundary between classes. For imbalanced datasets, this can be done through two families of solutions:

1. Extensive data processing: Operate on the dataset to modify data distribution.
2. Optimization of the classification algorithm: Operate on the learning algorithm to handle skewed class distribution in such a way that, examples with higher costs become harder to be misclassified.

Therefore, in our research, we apply methods from both families. More specifically, we will be interested in four methods that have been shown to be effective in handling the class imbalance problem, applied to cost-sensitive neural networks namely: over-sampling, under-sampling, class-weight, and threshold-moving. Our study reveals that these techniques are very effective.

3 Materials and Methods

In this section, we introduce our proposed method along with the used datasets. Our approach is composed of three main parts: data preparation and pre-processing, deep learning model, and cost-sensitive techniques. We use Amino-acid sequences for proteins and SMILES sequences as a drug representation. Hence, in the data processing phase, the drug and protein sequences were collected. The training drug-target datasets were then built for the following stage. Our deep learning model is based on convolutional recurrent stacked architecture. The cost-sensitive handling phase, is performed by using four methods: SMOTE over-sampling [31], Tomek links under-sampling [30], class-weight, and threshold moving. These techniques were applied to all the datasets.

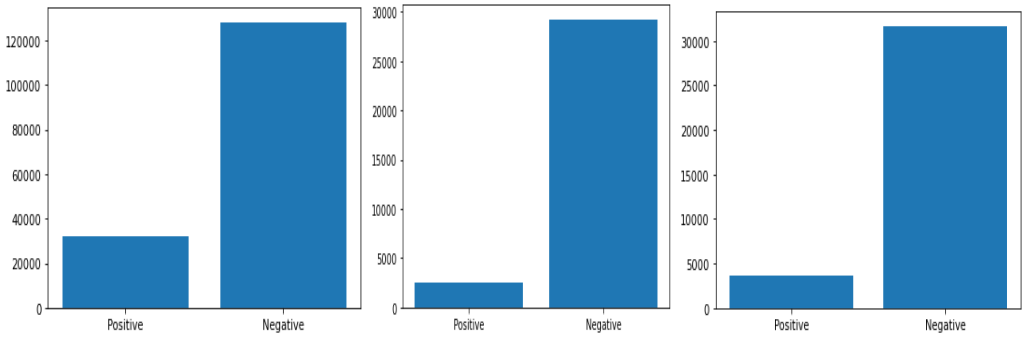
3.1 Datasets

To assess the performance of our approach, we use the three benchmark datasets: Davis [32], Metz [33], and KIBA [34]. Davis dataset contains selectivity assays with dissociation constant values. It contains interactions of 442 proteins and 68 ligands. KIBA dataset has been filtered to contain a total of 229 unique proteins

and 2116 unique drugs. Metz dataset consists of 1421 drugs and 156 targets. All these datasets contain binding affinities scores and were used in previous works on drug-target binding affinity prediction. Indeed, it was found that their use as binary datasets by defining a threshold for the score value, is more realistic than using datasets where all the unknown pairs interactions are considered as negative samples [35]. KronRLS [10], SimBoost [11], and Deepdta [36] are among the most known methods using these datasets. For their experiments, they used binarized versions of the datasets by applying thresholds. In this paper, we follow the same rules as done in these approaches. For the Metz dataset, we used the threshold of pKi $\geq$ 7.6 as suggested in [10] to assign a label of 1, i.e. the presence of interaction. For the Davis dataset, we used a threshold of pKd $\geq$ 7. In the KIBA dataset, [34] suggests a threshold of KIBA value $\leq$3.0 to binarize the dataset, we follow a similar approach. Table 1 summarizes the statistics for these three benchmark datasets. In this table, positive interaction represents the presence of interaction, while negative interaction reflects the absence of interaction. It is manifest that the datasets are highly imbalanced, as the number of negative samples is significantly larger than the positive samples, creating a bias issue. Table 1. presents a summary of the three datasets. In Fig. 1, the distribution of classes is illustrated. We notice the bias in all three datasets, with the highest degree of imbalance being in the Davis dataset.

Table 1. Datasets Summary

Dataset	Drugs	Proteins	Positive interaction	Negative Interaction	total	Positive class Rate (%)	Negative class Rate (%)
Davis	68	442	2562	29262	31824	8.05	91.95
Kiba	2116	229	32035	128261	160296	19.98	80.02
Metz	1421	156	3569	31690	35259	10.12	89.88



a- Kiba dataset b- Davis dataset c- Metz dataset

Fig. 1 - Classes Distribution for the three datasets: (a) Kiba dataset , (b) Davis dataset , (c) Metz Dataset

3.2 DRUG AND TARGET REPRESENTATION

Our model uses SMILES as drug representation, and amino-acid protein sequences for target representation. The Simplified Molecular-Input Line-Entry System (SMILES) is a specification in the form of a line notation that uses printable characters for describing the structure of chemical elements: molecules and reactions. SMILES is a true language, although with a small vocabulary size (atom and bond symbols) and only a few grammar rules. In our model, SMILES are represented by a one-hot encoding. Protein sequences are encoded similarly. Both SMILES and protein sequences have variable lengths, consequently, we opted for a maximal length of 1200 for proteins and 100 for drugs. The sequences that are shorter than the maximum length are 0-post-padded, while longer sequences are truncated. The choice of maximum length is guided by the length distribution of both sequences. We guaranteed that the chosen maximum lengths cover 95% of both proteins and compounds in the datasets. Our choice for SMILES as drug representation is motivated by the recent works in drug-target binding affinity prediction, that showed their supremacy even on graph-based methods [37].

3.3 BASE MODEL ARCHITECTURE

Our model is based on a deep convolutional recurrent architecture using CONVLSTM2D layers. This kind of layer combines the gating of LSTM with 2D convolutions. Input transformations and recurrent transformations are both convolutional. Convolutional networks are famous for their ability in discovering patterns, and LSTMs are a class of recurrent neural networks that excel in learning from sequential data. These characteristics motivated our use of CONVLSTM2D since, in our framework, both drugs and targets are represented by raw sequences that present patterns. . First a one-hot encoding is applied to each sequence. Then, an embedding layer is used to represent sequences characters with high-dimensional (128-dimensional) dense vectors. The resulting sequences are deeply explored by two convolutional recurrent blocks, each one composed of three CONVLSTM2D layers to capture the presence of discriminative features. These layers are directly connected without pooling which allows preserving the entire information, a pooling is performed at the end of the block, to reduce the output size of the previous layers and provide a generalization of the learned features. The output of the convolutional recurrent blocks are concatenated. To conclude, the final prediction is performed, in a standard way, by using fully connected layers after the feature extraction. All the hidden layers are activated by the “relu” activation function. The output layer was activated with the sigmoid () function. The whole neural network model was implemented with Keras [38,39]. The most powerful feature of CONVLSTM2D models is their ability to capture local dependencies with the help of filters and at the same time find out dependencies between distant sequence locations. The number and size of filters are among the

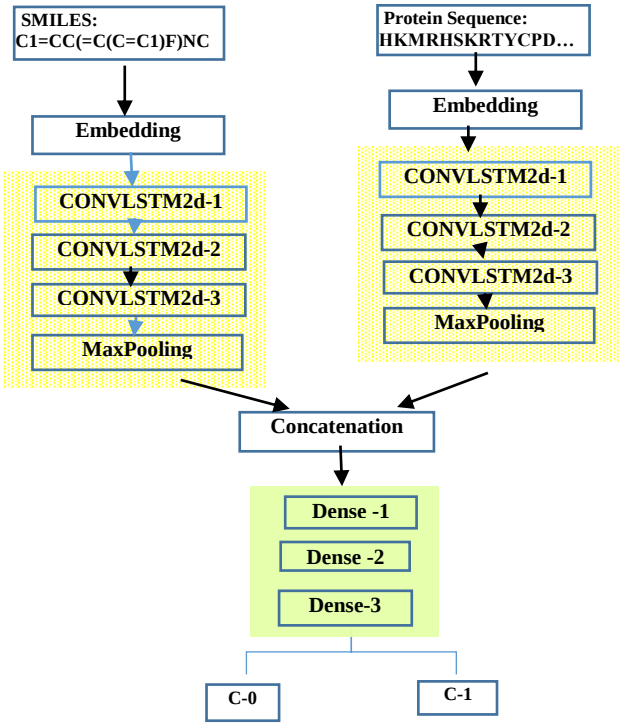


Fig 2. Base Model Architecture

hyperparameters that have a high impact on the model performance. Thus, increasing the number and the size of filters likely increases the ability of the model in patterns recognizing patterns [40]. The output layer was activated with the Sigmoid () function, which is the recommended activation for binary classification problems. The whole neural network model was implemented with Keras with tensorflow as backend [38,39]. The most powerful feature of CONVLSTM2D models is their ability to capture the local dependencies with the help of filters and at the same time find out dependencies between distant sequence locations. The number and size of the filters in a CONVLSTM2D has a direct impact on the features the model learns from the input. It is, therefore, well-established, that as the number of filters increases, the model becomes better at recognizing patterns. [40]. Stacking many CONVLSTM2D layers allows the automatic detection of more abstracted features. The numbers of filters we used and their sizes are described in th Table 2 below. For the ending fully connected block, we used 1024 nodes in the first two Dense layers, and 512 nodes in the last one. To overcome overfitting, we used dropout and batch normalization. Dropout is a technique that is used to avoid overfitting by excluding the activation of some of the neurons. Aggressive regularization is also employed to remove overfitting to the maximum extent. As a regularization technique, we use the L2- regularizer. The activation functions used for fully connected layers are all Rectified Linear Units (ReLU) which has been widely used in deep learning studies [41]. We used binary Crossentropy as a loss function. The learning was completed with 100

epochs and a mini-batch size of 128 was used to update the weights of the network. Adadelta optimizer [41] was used as the optimization algorithm to train the networks, with a Callback for initializing the learning rate and dynamically tuning it during the training by reducing its value based on the monitored parameter, the initial learning rate value was set to 0.01, and the minimum value was set to 0. 0001. The structure of the network is shown in Fig. 2

Table 2. Convolutional Recurrent Block layers

Layer	#Filters	Size	Dropout	Rec-dropout	Regularizer
Convlstm2d-1	16	4	0.2	0.1	L2
Convlstm2d-2	32	4	0.2	0.1	L2
Convlstm2d-3	64	4	0.2	0.1	L2

3.4 COST-SENSITIVE DTI

As shown by the datasets summary, the three datasets are exceedingly imbalanced and the classification samples are not uniformly distributed. In this study, we applied four classes of solutions. Two of these solutions, namely, over-sampling and under-sampling operate on datasets to modify the distribution of the training data such that the costs of the examples are conveyed explicitly by the appearances of the examples. While the two other methods: threshold moving and class-weight operate on the learning algorithm to handle the skewed class distribution, in such a way that examples with higher costs become harder to be misclassified. In general, cost-sensitive learning methods assign costs to classes. These costs are inversely proportional to the number of training examples of the corresponding class. So, in our study, the positive class will be assigned the highest cost. In this section, we present each of these methods.

3.4.1 Over-Sampling

Over-sampling operates on datasets; it changes the training data distribution in such a way that costs of the examples are concordant to examples occurrences. In other words, this method iteratively creates new instances of higher-cost until the number of different training examples are proportional to their costs. We applied the Synthetic Minority Oversampling Technique: SMOTE. This technique first chooses a positive instance (minority class instance) *xi* at random and finds its *k* nearest positive neighbors. Typically, and in our work *k=5*. A synthetic instance is then created by selecting one of the *k* nearest neighbors *xj* at random and performing an interpolation connecting *xi* and *xj* to form a line segment in the feature space. The synthetic instances are generated as a convex combination of the two chosen instances *xi* and *xj*. The newly generated instance value is given by the formula (1) below:

$$Xnew = xi + \lambda\,(xj - xi) \quad (1)$$

where λ is a random number in the range [0,1]. This interpolation creates a sample on the line between xi and xj. The procedure can be used to create as many synthetic examples for the positive class as are required. The approach is effective because new synthetic positive examples are created in a consistent way, that is, they are relatively close in feature space to existing examples of the positive class. We used the imbalanced-learning, imblearn [42] library to implement the SMOTE over-sampling.

.

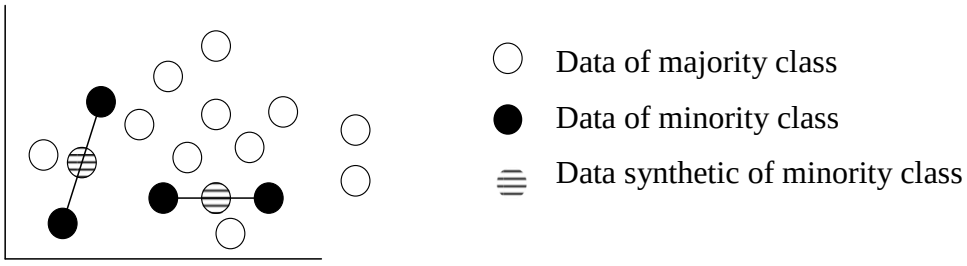


Fig.3 Illustration of the SMOTE technique

3.4.2 Under-Sampling

Under-sampling attempts to decrease the number of inexpensive examples, in our case, negative samples. We start by removing redundant examples at first and then removing borderline examples and examples suffering from the class label noise. The examples that are close to the limits between the two classes are called borderline examples. They are unreliable because even a small extent of noise or trouble can change their membership and move them to the opposite class. In this work, we used the Tomek links under-sampling method. It detects the borderline examples and examples suffering from the class label noise. The idea is to take two examples, i.e. *x* and *y*, such that each belongs to a different class, and then compute *Dist (x, y)* denoting the distance between them. The pair *(x, y)* is called a Tomek link if no example *z* exists such that:

$$Dist\,(x, z) < Dist\,(x, y) \text{ or } Dist\,(y, z) < Dist(y,x). \quad (2)$$

Two samples of different classes that are the nearest neighbors of each other produce a Tomek link. Consequently, we remove any observations from the majority class for which a Tomek link is detected. It removes unwanted overlap between classes where majority class links are removed until all pairs of closest neighbors, at minimum distance, are of the same class. Many studies have revealed that under-sampling is effective in learning with imbalanced datasets, sometimes even stronger than oversampling, especially on large datasets. We used the imblearn [42] library to implement Tomek link under-sampling.

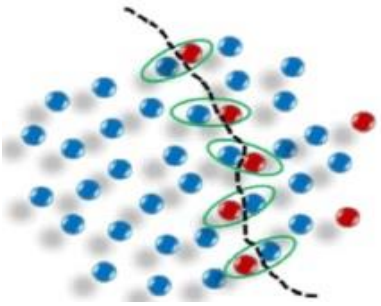


Fig.4 Illustration of Tomek Link technique

3.4.3 Threshold Moving

Our deep-learning model is predicting a probability of class membership which is interpreted before it can be mapped to a class label. This is accomplished by using a threshold, usually, 0.5, where all values equal to or greater than the threshold are mapped to one class, and all other values are mapped to another class. The default threshold results in poor performance for the classification problems suffering from highly imbalanced datasets. As such, we explore the idea of improving the performance of our model by tuning the threshold used to map probabilities to the class label. Threshold-moving moves the output threshold toward inexpensive classes such that examples with higher costs become harder to be misclassified. This method uses the original training set to train a neural network, the cost-sensitivity is introduced merely in the test phase. Recently, it has been recognized that "When studying problems with imbalanced data, using the classifiers produced by standard machine learning algorithms without adjusting the output threshold may well be a critical mistake" [43]. It has also been declared that trying other methods, such as sampling, without trying setting the threshold might be inappropriate [43]. A recent study has revealed that threshold moving is as effective as sampling methods in addressing the class imbalance problem [44]. Usually, threshold moving is performed by testing different threshold values and the resulting labels are evaluated using a selected evaluation metric. The threshold attaining the best evaluation metric is applied for the model. In our work, we use another method based on Youden's J-statistic [45] given by the formula (3) below, to define the optimal threshold value. The J-statistic attempts to maximize simultaneously the Sensitivity and Specificity of the model. This is achieved by maximizing their sum. These metrics are given by the formulas:

$$Sensitivity = True\ Positive\ Rate \quad (1)$$

$$Specificity = 1 - False\ Positive\ Rate \quad (2)$$

Thus, their sum is:

$$Sensitivity + Specificity = (\ True\ Positive\ Rate - False\ Positive\ Rate\ ) + 1$$

So, maximizing this sum implies simply maximizing the J-statistic given by:

$$J = TruePositiveRate - FalsePositiveRate \quad (3)$$

We then select the threshold with the largest J-statistic value. Accordingly, we performed the training of our initial model without threshold moving but by computing in parallel the optimal threshold as explained above. Then in the testing phase, the found value of the optimal threshold is used in evaluating new samples.

3.4.4 Class Weight

In cost-sensitive learning, class weight is one of the performant methods to overcome the class imbalance. One can integrate the weights of the classes into the cost function to make the classifier attentive to the imbalanced data. In our approach, we give a higher weight (cost) to the positive class and a lower weight to the negative class. This places more emphasis on the positive class such that the end result is a classifier that can learn equally from both classes. the positive class gains in importance as its errors are considered costlier than those of the negative class. For the training of our model, we used Crossentropy as a loss function since it is considered as the conventional cost function for binary classification algorithms. It is defined as:

$$Crossentropy = -ylog(p) - (1-y)log(1-p)$$

where y is the class binary indicator: 0 for negative class and 1 for positive class; p is the predicted probability for instance belonging to class 1. To incorporate the weights of the classes into the above formula, we define a weighted Crossentropy in which the weights are not the same for the two classes, but are based on the value ω for class 0 and (ω-1) for class 1. The weighted Crossentropy is then given by:

$$BalancedCrossEntropy = -\omega\,(ylog(p) - (1-\omega)(1-y)log(1-p)$$

ω is a given parameter that needs to be computed previously. We implemented the function that computes the optimal value of ω for each dataset based on its classes distribution.

4 Experiments and Results

In this section, we present and discuss the experimental results of our methods for DTIs prediction. We implemented our model in Python with the library tensorflow. keras [38,39]. We scrutinized the efficiency of the different techniques used in this research. Finally, we make a comparison of our suggested method against other similar methods.

4.1 Evaluation Metrics

Several performance metrics have been used to assess the performance of prediction approaches and compare classification models. The most used metrics are accuracy, precision, recall, and other similar metrics. Nevertheless, when the datasets are imbalanced, these metrics are not suitable to demonstrate the efficiency of a method. Usually, in similar situations, two metrics are insensitive to imbalanced ratio, namely, the area under the curve for Receiver Operating Characteristic (auROC) and the area under the Precision-Recall curve (auPR). These two metrics have been extensively used for imbalanced datasets as standard metrics for comparisons [46,47]. Both metrics value range from 0 to 1 where a random classifier has a score of 0.5 and a perfect classification model will have an auPR and auROC score of 1. In both cases the higher the better. Therefore, we have used these two effective measures to evaluate the classification performance of our method. A ROC curve is a plot of true positive versus false-positive rate for diverse threshold values. Another important factor is the bias and variance trade-off. Cross-validation is commonly used as an attempt to solve the bias-variance problem [48]. Among these cross-validation techniques, the k-fold cross-validation has been widely employed in research works since it always shows notable characteristics. The conventional value of k in DTI approaches is 5. Thus, we have implemented 5-fold shuffled cross-validation on each dataset. During the cross-validation procedure, the drug-target datasets are randomly separated into five distinct and non-overlapping folds of approximately the same size. Four folds are used as training samples, and the remaining fold is used as testing samples. The complete method is accomplished five times and the prediction results are calculated for each round. The final prediction result is the average of the cross-validation scores measures. For the oversampling SMOTE technique with the 5-fold cross-validation process, the original dataset is initially divided into training samples and testing samples. SMOTE is then applied to the training samples in each phase of cross-validation. So, merely the training samples are oversampled. the testing samples are not oversampled, and they are completely unseen during the training of the predictive model.

4.2 Baseline Methods

To assess the performance of our work, we compared it with three state-of-the-art methods, namely: DeepDTA [36], KronRLS [10], and SimBoost [11].

4.2.1 DeepDTA

DeepDTA, introduced in [36] is a deep-learning model based on Convolutional Neural Network (CNN) architecture that includes two distinct stacked CNN blocks, one for protein sequences and one for SMILES that learn separately latent features of drug and target, followed by pooling layers reducing the size.

DeepDTA performance was assessed on Kiba and Davis datasets. It was designed for continuous values prediction, but is also assessed for binary classification.

4.2.2 KronRLS

KronRLS is introduced in [10]. It is based on Regularized Least Squares Models (RLS) and it can predict both binary classes and continuous binding values. Given a set {di} of drugs and a set {tj} of targets, the goal of KronRLS is to learn a prediction function f(x) for all possible drug–target pairs $x \in \{di \times tj\}$. An objective function computing the dissimilarity between two pairs has to be learned. Thus, the problem of learning the prediction function f is formulated as finding a minimizer of a corresponding objective function [4].

4.2.3 SimBoost

SimBoost [11] is a gradient boosting machine-based method that, constructs feature for each drug, each target, and each drug–target pair. With each drug-target pair, SimBoost associates a feature vector. SimBoost uses three networks: drug-drug similarity network, target-target similarity network, and drug-target binding network. In the latter case, a node can either be a drug or target and the drug nodes and target nodes are connected to each other via binding affinity value. Latent vectors from matrix factorization are also included in this network. A supervised learning method named gradient boosting regression trees [49] is used for DTI prediction.

4.3 Results

In this section, we present the results obtained by our base model, which is merely the DL model without any imbalance treatment.; along with the results for each cost-sensitive technique and a comparison with previous methods. In the experiments reported in this paper, each dataset was split into two sets, train set and test set using 5-fold cross-validation. Table 3 and Table 4 present the results for our base model, respectively, in terms of Roc and AuPR metrics, with a comparison with baseline methods. The results for the baseline methods were taken from the experiments reported in the literature [10,11,36]. The performances of our imbalance treatment techniques, for the three datasets, are presented in Table 5 and Table 6.

Table 3. Performance of our base model on the three datasets in terms of AUROC with comparison to baseline methods.

	Davis	Kiba	Metz
KronRLS	0.931	0.904	0.932
SimBoost	0.956	0.907	**0.958**
DeepDta	N/A	N/A	N/A
Our base Model	**0.959**	**0.934**	0.929

Table 4. Performance of our base Model on the three datasets in terms of AUPR with comparison to baseline methods.

	Davis	Kiba	Metz
KronLRS	0.686	0.766	0.565
SimBoost	0.758	0.782	0.629
DeepDta	0.714	**0.788**	N/A
Our base Method	**0.772**	0.760	**0.6341**

We notice that our base model shows high performances in both ROC and AUPR, even if it is not the best in all the datasets compared to the baseline methods. Tables 3 and 4 show that our method is able to produce results with the best AUROC and AUPR in two datasets out of 3. Still, for the dataset, where the performance of our method is not the best, it is very close to the best performing. This could be explained by the pertinence of the choice of CONVLSTM2D architecture, and the adequacy of the hyper-parameters tuning.

Table 5. Performance of our unbalancing methods on the three datasets in terms of auROC

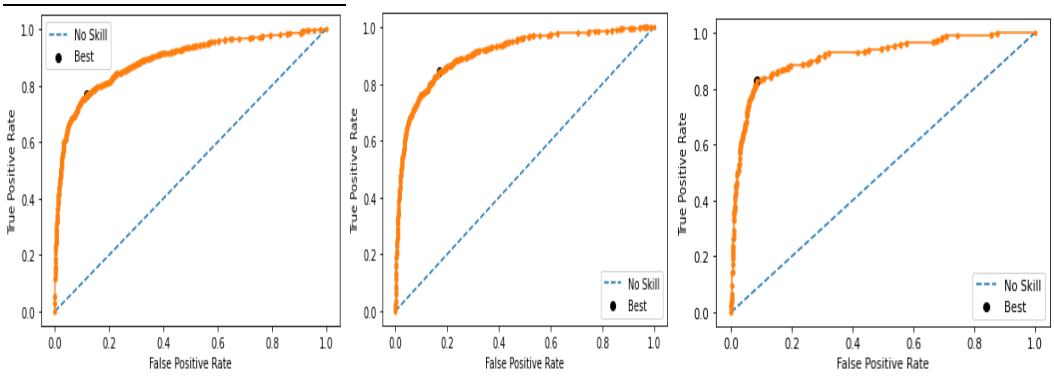
Methods	Davis	Kiba	Metz
SMOTE	0.978	0.965	0.964
TOMEK LINK	0.907	0.875	0.883
CLASS-WEIGHTS	0.974	0.955	0.960
THRESHOLD MOVING	0.963	0.937	0.953

Table 6. Performance of our unbalancing methods on the three datasets in terms of auROC

Methods	Davis	Kiba	Metz
SMOTE	0.8482	0.805	0.864
TOMEK LINK	0.668	0.703	0.582
CLASS-WEIGHTS	0.827	0.796	0.7072
THRESHOLD MOVING	0.795	0.774	0.649

The imbalance handling techniques show overall a great improvement of the results. The SMOTE over-sampling was very effective in all the datasets. We noticed two unexpected results. The first is about the great efficiency of Class-weighing, indeed the results show that when we find the optimal class weights, this method is very effective. The second remark is about the Tomek link under-

sampling method, which is usually known to be efficient but did not work very well for our experiments, its results could be explained by the complexity of our learning problem which requires larger datasets; therefore, the results were seriously affected by the reduction in the size of the datasets. We can consider a simple ad-hoc solution of this variation, by following this method directly; by SMOTE which will catch up with the reduced number of instances by creating new ones. The threshold moving method also improved the result of our base model, the Fig. 5 presents the AuRoc for the three datasets, with the optimal threshold value for each one. Thus, for both metrics, for all the datasets, all our methods, except under-sampling, have either the best performance, or their performance is very close to the best one.



(a) Metz Dataset (b) Kiba Dataset (c) Davis Dataset

Fig. 5 AUROC for the three datasets with the optimal threshold at the black point at the top- left of the plot.

5 Conclusion

To tackle the problem of DTI prediction more efficiently, we propose a cost-sensitive deep learning based method. We represent both drugs and proteins with sequences. Then we use convolutional recurrent Neural Networks (CONVLSTM2D) to learn representations from the combined drug-target sequence. Our results showed that the use of stacked CONVLSTM2D to learn representations of proteins and drug sequences is appropriate. This could be an indication that compound and amino-acid sequences require a structure that can handle simultaneously, their ordered relationships along with their hidden patterns, which the CONVLSTM architecture accomplished successfully. We perform our experiments on three commonly used datasets, Davis, Kiba, and Metz. We implemented various methods to deal with the increased imbalance rate

of the conventional datasets. We evaluated our methods by using auROC and auPR metrics, which are the most suitable metrics when the datasets are imbalanced. In this work, we used four balancing methods including under-sampling Tomek Link, over-sampling SMOTE, Threshold moving, and Class-weight. As a result, our methods have revealed benefits to deal with the bias issue and provided better prediction performance for the three datasets. Our proposed method outperforms other state-of–that-art methods in terms of auPR and auROC metrics. The results suggest that cost-sensitive learning is effective for an unbalanced binary classification. It also reveals that using very complicated forms for entry is not the ultimate way to tackle the complexity of DTI detection. Instead, using a simpler input representation and approaching the problem differently has actually shown better results. It is also very reasonable to use these methods to further improve the results, we can easily combine our techniques. As example, combine a sampling method each time, with a threshold moving or a class weight. We can also start with under-sampling to eliminate noisy and borderline samples, and then follow up with over-sampling to create more reliable samples. It is also conceivable to try other improvements by deepening the network, but at the cost of more computation costs, and with the risk of overfitting.

References

[1] J.Thafar M, Raies AB, Albaradei S, Essack M and Bajic VB (2019) Comparison Study of Computational Prediction Tools for Drug-Target Binding Affinities. Front. Chem. 7:782. doi: 10.3389/fchem.2019.00782

[2] D.-S. Cao et al.,``Large-scale prediction of drug_target interactions using protein sequences and drug topological structures," Anal. Chim. Acta, vol. 752, pp. 1_10, Nov. 2012.

[3] L. Yang et al.,``Exploring off-targets and off-systems for adverse drug reactions via chemical-protein interactome_clozapine-induced agranulocytosis as a case study," PLoS Comput. Biol., vol. 7, no. 3, 2011, Art. no. e1002016

[4] M. J. Keiser, B. L. Roth, B. N. Armbruster, P. Ernsberger, J. J. Irwin, and B. K. Shoichet, ``Relating protein pharmacology by ligand chemistry," Nature Biotechnol., vol. 25, no. 2, pp. 197_206, 2007

[5] Y. Yamanishi, M. Araki, A. Gutteridge, W. Honda, M. Kanehisa, Prediction of drug–target interaction networks from the integration of chemical and genomic spaces, Bioinformatics 24(13) (2008), i232–i240.

[6] Z. Mousavian, S. Khakabimamaghani, K. Kavousi, and A. Masoudi-Nejad, ``Drug-target interaction prediction from PSSM based evolutionary information," J. Pharmacol. Toxicol. Methods, vol. 78, pp. 42_51, Mar./Apr. 2016.

[7] Z. He et al., ``Predicting drug-target interaction networks based on functional groups and biological features," PLoS ONE, vol. 5, no. 3, 2010, Art. no. e9603.

[8] X. Xiao, J.-L. Min, P. Wang, and K.-C. Chou, ``iCDI-PseFpt: Identify the channel_drug interaction in cellular networking with PseAAC and molecular fingerprints," J. Theor. Biol., vol. 337, pp. 71_79, Nov. 2013.
[9] H. Yu et al., ``A systematic prediction of multiple drug-target interactions from chemical, genomic, and pharmacological data," PLoS ONE, vol. 7, no. 5, 2012, Art. no. e37608.
[10] Pahikkala,T. et al. (2014) Toward more realistic drug–target interaction prediction Brief. Bioinformatics, vol. 16, no. 2, pp. 325_337, 2014

[11] Simboost He,T. et al. (2017) Simboost: a read-across approach for predicting drug–target binding affinities using gradient boosting machines. J. Cheminform., 9, 24
[12] Y. Yamanishi, M. Kotera, M. Kanehisa, and S. Goto, ``Drug-target interaction prediction from chemical, genomic and pharmacological data in an integrated framework," Bioinformatics, vol. 26, no. 12, pp. i246_i254, 2010.
[13] Z. Li et al., ``In silico prediction of drug-target interaction networks based on drug chemical structure and protein sequences," Sci. Rep., vol. 7, no. 1, 2017, Art. no. 11174.
[14] M. Hao, Y. Wang, and S. H. Bryant, ``Improved prediction of drug-target interactions using regularized least squares integrating with kernel fusion technique," Anal. Chim. Acta, vol. 909, pp. 41_50, Feb. 2016.
[15] M. Gönen, ``Predicting drug-target interactions from chemical and genomic kernels using Bayesian matrix factorization," Bioinformatics, vol. 28, no. 18, pp. 2304_2310, 2012.
[16] J.-Y. Shi, A.-Q. Zhang, S.-W. Zhang, K.-T. Mao, and S.-M. Yiu, ``A uni_ed solution for different scenarios of predicting drug-target interactions via triple matrix factorization," BMC Syst. Biol., vol. 12, p. 136, Dec. 2018.
[17] Y. Liu, M.Wu, C. Miao, P. Zhao, and X.-L. Li, ``Neighborhood regularized logistic matrix factorization for drug-target interaction prediction," PLOS Comput. Biol., vol. 12, no. 2, 2016, Art. no. e1004760
[18] W. Ba-Alawi, O. Soufan, M. Essack, P. Kalnis, V.B. Bajic, Daspfind: new efficient method to predict drug–target interactions, J. Cheminform. 8(1) (2016) 15.
[19] LeCun, Y. Bengio, and G. Hinton. Deep learning. nature, 521(7553):436-444 (2015)
[20] Leung,M.K. et al. Deep learning of the tissue-regulated splicing code. Bioinformatics, 30, 2014, i121–i129.
[21] Xiong,H.Y. et al. The human splicing code reveals new insights into the genetic determinants of disease. Science, 347, 2015, 1254806.
[22] Ma,J. et al. Deep neural nets as a method for quantitative structure–activity relationships. J. Chem. Inf. Model., 55, 2015, pp.263–274.
[23] Jing, Y., Bian, Y., Hu, Z.,Wang, L., and Xie, X.-Q. S. (2018). Deep learning for drug design: an artificial intelligence paradigm for drug discovery in the big data era. AAPS J. 2018, 20:58. doi: 10.1208/s12248-018-0210-0

[24] Ekins, S., Puhl, A. C., Zorn, K.M., Lane, T. R., Russo, D. P., Klein, J. Exploiting machine learning for end-to-end drug discovery and development Nat. Mater. 18, 2019, pp.435–441. doi: 10.1038/s41563-019-0338-z et al.

[25] K.C. Chan, Z.-H. You, et al., Large-scale prediction of drug-target interactions from deep representations, in: 2016 International Joint Conference on Neural Networks, IJCNN, IEEE, IEEE, 2016, pp.1236–1243.

[26] K. Tian, M. Shao, Y. Wang, J. Guan, S. Zhou, Boosting compound-protein interaction prediction by deep learning, Methods 110 (2016) 64–72

[27] L. Wang, Z.-H. You, X. Chen, S.-X. Xia, F. Liu, X. Yan, Y. Zhou, K.-J. Song, A computational-based method for predicting drug–target interactions by using stacked autoencoder deep neural network, J. Comput. Biol. 25(3) (2018) 361–373.

[28] M. Wen, Z. Zhang, S. Niu, H. Sha, R. Yang, Y. Yun, H. Lu, Deep-learning-based drug–target interaction prediction, J. Proteome Res. 16(4) (2017) 1401–1409.

[29] Jtastrzkeski,S. et al. Learning to smile (s). arXiv preprint arXiv: 1602.06289. (2016)

[30] I. Tomek, "Two modifications of CNN," IEEE Transactions on Systems, Man and Cybernetics, vol.6, no.6, pp.769–772, 1976.

[31] N. V. Chawla, K.W. Bowyer, L. O. Hall, andW. P. Kegelmeyer, ``SMOTE: Synthetic minority over-sampling technique," J. Artif. Intell. Res., vol. 16, no. 28, pp. 321 357, Jun. 2006.

[32] Davis,M.I. et al. (2011) Comprehensive analysis of kinase inhibitor selectivity. Nat. Biotechnol., 29, 1046–1051.

[33] J. T. Metz, E. F. Johnson, N. B. Soni, P. J. Merta, L. Kifle, and P. J. Hajduk. Navigating the kinome. Nat. Chem. Biol., 7(4):200, 2011.

[34] Tang,J. et al. (2014) Making sense of large-scale kinase inhibitor bioactivity data sets: a comparative and integrative analysis. J. Chem. Inf. Model., 54, 735–743.

[35] F. Provost, "Machine learning from imbalanced data sets 101," in Working Notes of the AAAI'00 Workshop on Learning from Imbalanced Data Sets, Austin, TX, pp.1–3, 2000.

[36] Öztürk, H., Özgür, A., and Ozkirimli, E. DeepDTA: deep drug-target binding affinity prediction. Bioinformatics 34 2018, i821–i829. doi: 10.1093/bioinformatics

[37] Padme Feng, Q. (2019). PADME: A Deep Learning-based Framework for Drug-Target Interaction Prediction (Master thesis), Simon Fraser University, Burnaby, BC, Canada.

[38] Chollet,F. et al. (2015) Keras. https://github.com/fchollet/keras.

[39] Abadi,M. et al. (2016) Tensorflow: a system for large- learning. In: OSDI scale machine, Vol. 16, pp. 265–283.

[40] Kang,L. et al. Convolutional neural networks for no-reference image quality assessment. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Washington, DC, USA, 2014, pp. 1733–1740. [52].

[41] Goodfellow I., Bengio Y., Courville A. Deep Learning. MIT Press, http://www.deeplearningbook.org 2016.
[42] https://github.com/scikit-learn-contrib/imbalanced-learn
[43] M.A. Maloof, “Learning when data sets are imbalanced and when costs are unequal and unknown,” in Working Notes of the ICML’03 Workshop on Learning from Imbalanced Data Sets, Washington, DC, 2003.
[44] Zhi-Hua Zhou and Xu-Ying Liu. Training cost-sensitive neural networks with methods addressing the class imbalance problem. IEEE Transactions on Knowledge and Data Engineering, 18(1):63–77, 2006.
[45] Powers, David M W (2011). "Evaluation: From Precision, Recall and F-Score to ROC, Informedness, Markedness & Correlation". Journal of Machine Learning Technologies. 2 (1): 37–63. hdl:2328/27165.
[46] N. Japkowicz, “Learning from imbalanced data sets: a comparison of various strategies,” in Working Notes of the AAAI’00 Workshop on Learning from Imbalanced Data Sets, Austin, TX, pp.10–15, 2000.
[47] N. Japkowicz and S. Stephen, “The class imbalance problem: a systematic study,” Intelligent Data Analysis, vol.6, no.5, pp.429–450, 2002.
[48] M. S. Santos, J. P. Soares, P. H. Abreu, H. Araujo, and J. Santos, ``Crossvalidation for imbalanced datasets:Avoiding overoptimistic and overfitting approaches [research frontier]," IEEE Comput. Intell. Mag., vol. 13, no. 4, pp. 59 76, Nov. 2018.
[49] H. Chen, Z. Zhang, A semi-supervised method for drug-target interaction prediction with consistency in networks, PLoS ONE 8(5) (2013) e62975