Drug–target interaction prediction with a deep-learning-based model

Lingwei Xie+, Zhongnan Zhang*
Software School, Xiamen University
Xiamen, China, 361005
*zhongnan_zhang@xmu.edu.cn

Song He+, Xiaochen Bo*, Xinyu Song
Beijing Institute of Radiation Medicine
Beijing, China, 100850
*boxiaoc@163.com

***Abstract*—Drug-target interaction identification is of highly importance in drug research and development. The traditional experimental paradigm is costly, while the previous *in silico* prediction paradigm remains a challenge because of diversified data production platforms and data scarcity. In this paper, we modeled drug-target interaction prediction as a binary classification task based on transcriptome data of drug stimulation and gene knockout from LINCS project and developed a framework with a deep-learning-based model to predict potential interactions. The evaluation results showed that not only did our framework fit data with better accuracy than other classical methods, but predicted more credible drug-target interactions. What's more, the prediction has high percentage of overlap interactions across other platforms.**

Keywords-drug-target interaction; deep learning; LINCS projetct; transcriptome data

I. INTRODUCTION

The identification of interactions between drugs and target proteins, a critical to drug research and development, can be used in the fields of drug repositioning and side-effect prediction [1], [2]. Amounts of drug–target interactions (DTIs) are uncovered and stored in databases, including Drugbank, TTD, and Matador, but certain DTIs remain to be discovered [3-5]. Although the high-throughput screening technology is available, the traditional experimental strategy of DTI discovery is still costly and time consuming.

The *in silico* prediction using a variety of computational models are developed to address the problem. Campillos et al. developed an algorithm to infer whether two drugs have the same target protein based on side effect similarity [6]. Bleakley et al. proposed a framework, named Bipartite Local Model, by modeling unknown DTI edges as a binary classification problem [7]. Wang et al. developed a machine learning algorithm to predict DTIs by introducing the framework of restricted Boltzmann machines [8]. Yamanishi et al. proposed a bipartite graph learning method to predict DTIs by integrating chemical and genomic spaces [9]. In principle, the performance of DTI prediction depends on both data source and prediction algorithm. In addition to cellular response data, researchers tried using side effect data, chemical data, and pharmacological data to identify the novel interactions between drugs and target proteins [6], [10], [11]. However, the performance of *in silico* prediction remains a challenge because of diversified data production platforms and data scarcity. The National Institute of Health (NIH) proposed the Library of Integrated Network-based Cellular Signatures (LINCS) project in 2010. This project intends to describe a comprehensive picture of multilevel cellular response when cells are exposed to a variety of perturbing environments, including drug stimulation and gene knockout (http://www.lincscloud.org/l1000/). The L1000 database of LINCS project includes millions of genome-wide expression profiles when 72 cell lines are stimulated by more than 20,000 small molecule compounds or when more than 4,000 genes are respectively knocked out in these cell lines. It provides a unified and extensive transcriptome data source for DTI prediction.

In this paper, the deep learning method is regarded as a tool for discovering potential relationships between drugs and target proteins because of the increasing availability of big data and GPU computing, and the rapid development of deep learning. Hence, we explore the possibility of deep neural network (DNN) to predict new DTIs based on transcriptome data of drug perturbation and gene knockout trails from the L1000 database. Specially, we propose a framework for the combination of drug data and gene data, sampling from negative sample space, training a DNN we designed, and DTI prediction. This framework is inspired by the intrinsic nonlinear patterns, which possess better prospect for inference, within the LINCS project [12]. First, we made a permutation and combination of gene expression data from drugs and genes, both from L1000 database, in a serial manner according to known DTIs in the Drugbank database [3]. Second, all positive samples and distributed negative samples formed input space for training and evaluating a DNN model that only has 2,000 hidden units. In forward propagation, each feature dimensionality of a sample was reduced approximately 200 times. When training finished, the DNN model derived a decision boundary to classify positive and negative samples with desired accuracy and predicted reliable DTIs. Lastly, the predicted results were more analyzed by distance metric (D-score) and cross-platform comparisons. Further validation illustrates that our framework can predict a certain part of novel DTIs validated by known experiments in other databases, including TTD, MATADOR, IUPHAR/BPS, and STITCH [4], [5], [13], [14]. These results prove that the DNN model we designed is capable of extracting low dimensional features to represent raw dataset effectively and classifying positive and negative samples accurately. Furthermore, our framework has the

*To whom correspondence should be addressed.

+These authors contributed equally to this work and should be considered as co-first authors.

ability to integrate transcriptome data from drugs and genes, and possess potential and wider prospect for DTI prediction for improving the drug discovery process.

II. METHODS

In this section, how to discover new DTIs is discussed, including data from L1000 and DrugBank databases, problem definition, and supervised learning.

In this study, we modeled the problem to explore unknown DTIs as a binary classification task in machine learning domain. First of all, considerable expression data, under various drug perturbation and gene knockout trails, were selected as the original dataset, and some genes were target proteins while others were not. However, the resulting number of negative data is far more than the positive data in PC3 cell line. The input space consisted of all positive data and a part of presentative expression data that were uniformly sampled from negative sample space. Then, the feature space was from combination of expressions of drugs and genes. Lastly, the model was used for DTI prediction after fitting training data with high accuracy.

A. Data from the L1000 database

The LINCS project proposed in 2010 intends to create a network-based understanding of multilevel cellular changes when cells are exposed to a variety of perturbing environments and deciphers how cells respond to a variety of genetic and chemical stresses. The pilot phase of the project was accomplished in 2013, and generated more than 660,000 gene expression profiles of drug perturbation and more than 440,000 gene expression profiles of gene knockout perturbation.

The L1000 biotech applied in the project is a new technology, measuring the expression of only 978 landmark genes and using the correlation of the gene to infer the remaining ~20,000 gene expressions. The data structure of this project, like the TCGA project (https://cancergenome.nih.gov/), consists of four levels. Level 1 data refers to the expression value of 978 landmark genes, while level 2 data refers to the normalized expression value of 978 landmark genes. Level 3 data recorded genome-wide gene expression, while level 4 data recorded the Z-score of genome-wide gene expression. In this paper, we use level 4 data of drug perturbation and gene knockout perturbation in the PC3 cell line.

Few perturbation names in LINCS can map to the approved drugs in Drugbank because the project is still ongoing. We select level 4 data of 480 FDA-approved drugs perturbation and 4,363 genes knockout perturbation in the PC3 cell line. We use the Z-score of only 978 landmark genes to reduce the feature dimension.

For all trials of a certain drug or a gene, we first calculated their Pearson correlation coefficient matrix. Then, we used the *k*-means method to divide them into several groups, and chose a group with the maximum intra-class Pearson correlation coefficient as the representation of this drug or gene, denoted by S_1. Meanwhile, to retain more information of these trials of the drug or gene, we averaged all trials data as an independent sample S_2. Lastly, we constructed a credible set S of this drug using S_1 and S_2.

B. DTI database

We trained and tested our model by using DTIs in the Drugbank database, a comprehensive drug data source, recording chemical, pharmacological, and pharmaceutical feature of over 8,000 drugs, including 2016 FDA-approved drugs [3]. In this paper, we use version 5.0 of the Drugbank database. To make cross-platform comparisons compatible, we take the PubChem ID as the identifier of drugs across Drugbank and LINCS database. We finally use 918 interactions between 415 drugs and 350 targets from the Drugbank database.

In addition, we validated DTIs between 623 drugs and 378 targets predicted by our model on four datasets derived from TTD, MATADOR, IUPHAR/BPS, and STITCH. For 632 drugs, we selected 2,529 interactions from TTD, 15,843 interactions from MATADOR, 13,679 interactions from IUPHAR/BPS, and 3,424 interactions from STITCH.

C. Problem Definition

In this work, the relationships between drugs and target proteins, both from L1000, were provided by DrugBank. The proposed approach implied that input consisted of two data channels, drug data, and gene data, respectively, and output can be represented by a binary random variable. Our intention was to discover new DTIs. For our work, DTI prediction was modeled as a binary classification task. Each sample was constructed by fusing a drug data and a gene data. The definition details in this task are as follows:

Definition 1: Drug matrix DM is an m by n matrix that contains all data samples of the drug set. m is the number of drug samples, and n is the number of drug features. Each line represents one drug.

Definition 2: Gene matrix GM is a q by n matrix that contains all data samples of gene set. q is the number of gene samples, and n is the number of gene features. Each line represents one gene.

Definition 3: Feature $DM_{i,j}$ or $GM_{i,j}$ is a real number that corresponds to the expression of the jth locus for sample i.

Definition 4: Label matrix LM is a q by m matrix. $LM_{i,j}$ is one label for interaction between gene i and drug j. If $LM_{i,j}$ is 1, then the combination of drug j and gene i (also suggest target protein i) is a positive sample; otherwise, the combination of drug j and gene i is either unlabeled sample or negative sample, depending whether gene i is one of target proteins or not.

Definition 5: Classification matrix CM is an l by k (k equals 2 in this work) matrix. $CM_{i,0}$ is the probability of sample i belonging to negative class. $CM_{i,1}$ is the probability of sample i belonging to positive class.

D. Supervised Learning

Hypothesis space F is the set of joint probability distributions and conditional probability distributions. In supervised learning, because infinite models are presented in the hypothesis space, if model f is selected as a decision function, for any input X, the predicted value $Y^*=f(X)$ is

obtained. The objective function, a real-valued function of $f(X)$ and Y, is constructed for evaluating the accuracy of training and defined by $L(Y, f(X))$, because L measures the nearness of the predicted value to the true value. As the loss value of the object function becomes smaller, the model fits better on training sets.

1) Logistic Regression

Logistic Regression (LR) is a better solution for predicting binary-valued labels ($Y^* \in \{0, 1\}$) than linear function $Y^*=f_\theta(x)= \theta^T x$ alone. The hypothesis class of LR, as defined in (1), tries to predict probability that a given sample belongs to the positive class versus the probability that it belongs to the negative class. For each training sample x, the following loss function, as defined in (2), measures how well a given f_θ does, and L2 penalty is introduced for regularization purpose.

$$P(y|x) = f_\theta(x) = \frac{1}{1+\exp(-\theta^T x)} \tag{1}$$

$$(\theta, b) = \arg\min_{\theta,b} \sum_{x \in X} -y \ln(f_\theta(x)) - (1-y)\ln(1-f_\theta(x)) + \lambda \|\theta\|_2 \tag{2}$$

2) Support Vector Machine and Random Forest

For binary classification tasks in the machine learning domain, we employed two methods that are widely used: Support Vector Machine (SVM) and Random Forest (RF). As a discriminative model, SVM builds a high-dimensional (even infinite dimensional) hyperplane $\theta \bullet x+b=0$, which is called the decision boundary for classifying samples, and models conditional probability distribution $P(Y|X)$ directly through optimizing hinge loss function, as defined in (3). The support vectors refer to the sample points that are closest to the decision boundary, so the bigger the margin, the better SVM fits the training sets. Furthermore, kernel functions are a class of algorithms for pattern analysis, whose best known member is the SVM. In this work, the algorithm can have selected different kernels (linear, RBF, and multinomial) for projecting data into different feature spaces. Eventually, multinomial, which can transfer data to approximate linear separable space and help to improve the performance of SVM, was indeed the most preferred. However, the computation for training a SVM classifier was expensive because the training was not parallel. In PC3 cell line that contains large amount of trails, time-consuming training is unacceptable over data increasing.

$$(\theta, b) = \arg\min_{\theta,b} \sum_{x \in X} \left[1 - y(\theta \bullet x + b)\right] + \lambda \|\theta\|_2 \tag{3}$$

To maintain tradeoff between efficiency and performance in practical tasks, ensemble learning combining many weak learner was adopted for classification. RF—an ensemble classifier that combines decision trees grown on random input vectors, and splits nodes on a random subset of features—is recognized as a robust classifier. When 100 decision trees exist, the performance of RF is close to or better than SVM [15]. Not only does RF eliminate the disadvantage of instability for only the decision tree, but also has the capacity to cope with high-dimensional feature space. In fact, feature selection is implicitly incorporated during each tree construction, resulting from each node of a decision tree, the best variable to split on a random subset of variables is selected. During classification, only those features needed for the test pattern under consideration are involved [16]. Resulting from each decision tree could be constructed respectively by minimizing the gini index, as defined in (4)–(5), and the learning process was accelerated in parallel.

$$\mathrm{Gini}(D) = 1 - \sum_{i=1}^{k} \left(\frac{|C_k|}{|D|} \right)^2 \tag{4}$$

$$\mathrm{Gini}(D, A) = \frac{|D_1|}{|D|}\mathrm{Gini}(D_1) + \frac{|D_2|}{|D|}\mathrm{Gini}(D_2) \tag{5}$$

Where D is whole training set, C_k denotes the number of samples belongs to class k in D, A denotes one of features with minimum gini index in D, and D_i is one of branches of D after split on feature A.

3) DNN

DNN functioned as an effective tool for DTI prediction, owing to modern biology entering the era of big data. The computing power was derived through, first, massively parallel distributed structure and, second, the ability to learn and generalize. DNN had a built-in capability to adapt their weights according to the changes of the surrounding environment. In particular, a neural network trained to operate in a specific environment can easily be retrained to handle minor changes in the operating environmental conditions.

Every neuron in DNN was nonlinear, which is a highly important property, particularly if the underlying physical mechanism responsible for generation of the input signal is inherently nonlinear, and potentially affected by the global activity of all other neurons in the network. Above of all, DNN extracted high-level features representing raw biological and chemical data automatically. Although DNN can deal with high-dimensional gene-expression data, the architecture was still a huge challenge. In this study, we modeled DTI prediction as a binary classification task to discover potential relationships between drugs and target proteins. Therefore, the input layer consisted of two channels for taking drug data and gene data from L1000 database as input, and the output layer contained only two neurons for binary classification that indicated the effectiveness of the drug to the target protein. The depth and breadth for hidden layers affected the performance of the network; if they were too large, the risk of over-fitting increased, otherwise, the performance is lost. We investigated the optimal number of hidden layers, number of hidden neurons, dropout rejection rate, and class imbalance weight by K-fold cross-validation. Lastly, Softmax, as defined in (6), was adopted for binary classification layer.

$$f_\theta\left(x^{(i)}\right) = \begin{bmatrix} P\left(y^{(i)}=1 \middle| x^{(i)}; \theta_1\right) \\ P\left(y^{(i)}=2 \middle| x^{(i)}; \theta_2\right) \\ \cdots \\ P\left(y^{(i)}=q \middle| x^{(i)}; \theta_k\right) \end{bmatrix} = \frac{1}{\sum_{j=1}^{k} e^{\theta_j^T x^{(i)}}} \begin{bmatrix} e^{\theta_1^T x^{(i)}} \\ e^{\theta_2^T x^{(i)}} \\ \cdots \\ e^{\theta_k^T x^{(i)}} \end{bmatrix} \tag{6}$$

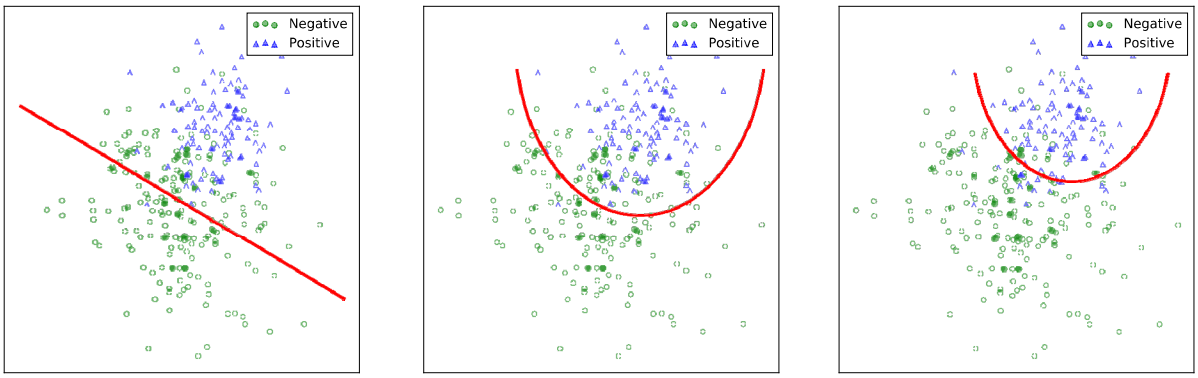


Figure 1. The decision boundary of DNN: The DNN fits training data with nonlinear decision boundary instead of hyperplane in high dimensional space, and the final decision boundary approximates positive cluster iteratively during model training, even sacrificing a bit of validation accuracy.

In the training procedure, each layer was randomly initialized first; each neuron was activated by ReLU with strong biological stimulation and mathematical justification. Training was completed by AdamOptimizer to minimize the loss value from objective function that is a cross entropy cost function followed by L1 penalty for all negative samples' probabilities of belonging to the negative class, as defined in (7), under the ratio of 1:2 (positive to negative). Although such operation sacrifices a bit of accuracy, the trained model has better potential for DTI prediction. As shown in Fig. 1, DNN fits training data with a nonlinear decision boundary (middle plot) rather than hyperplane (left plot). Furthermore, the ratio of positive to negative provide more information to make network learn features of negative samples, and the rebuilt objective function also paid more attention to real negative class for punishing false positive samples to push the decision boundary to be closer to the center of the positive class cluster (right plot).

$$(\theta,b)=\arg\min_{\theta,b}\sum_{x\in X}-y\ln\left(f_\theta(x)\right)-(1-y)\ln\left(1-f_\theta(x)\right)+\eta\left\|CM_{x,0}\right\|_1 \quad (7)$$

The trained model was measured on validation sets with other methods by validation accuracy, F-score, and predictive error (as defined in (8)) at each sample x_i. Every new interaction was measured by the value of $CM_{x,1}$, and further analyzed through the distance from the sample to decision boundary, as defined in (9). Such distance function was inspired by converting distance function to probability in a tree kernel-based SVM [17], and for binary classification the hypothesis of Softmax is equivalent to SVM.

$$\mathrm{PE}(x_i)=CM^{\text{other}}_{x_i,y_i}-CM^{\text{DNN}}_{x_i,y_i} \quad (8)$$

$$\text{D-score}(f\mid X)=\ln\left(\frac{CM_{X,1}}{1-CM_{X,1}}\right) \quad (9)$$

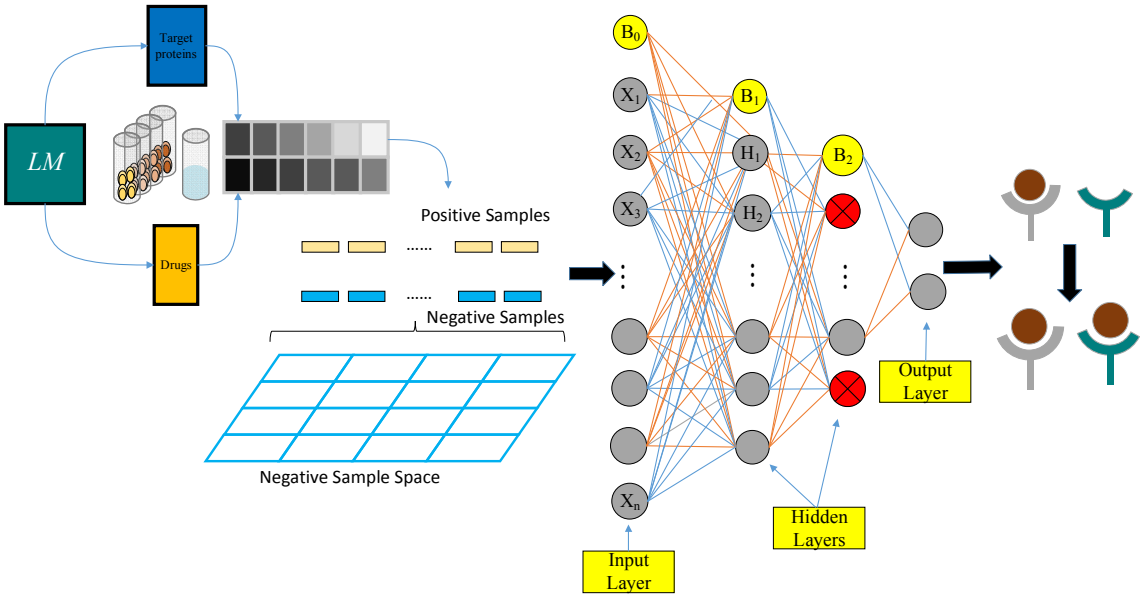


Figure 2. The whole process of the framework contains feature fusion, negative data sampling, model training, and prediction.

TABLE I. OVERALL PERFORMANCE OF DNN WITH DIFFERENT ARCHITECTURES

	Number of parameters of two hidden layers				
	≤1,000	≤4,000	≤10,000	≤1,000,000	>1,000,000
Number of hidden neurons	[10:100, 10]	[200, 2:21]	[500:1,000, 10]	[1,000:1,500, 20:600]	[1,000:2,900, 110:2,900]
	89.34% ± 2.06	90.02% ± 4.54	89.57% ± 1.17	87.92% ± 1.71	88.67% ± 1.57

III. RESULTS

In this work, the discovery of new DTIs was modeled as a binary classification task. The whole dataset contains all expression data of drugs, target protein genes, and non-target protein genes. However, the number of positive samples (combination of drug data and target protein gene data) is lesser than the number of negative samples (combination of drug data and non-target protein gene data). Therefore, the input space consists of all positive samples and uniformly sampled negative samples. Resulting from some intrinsic linear and nonlinear patterns in LINCS project [12], we took LR to capture linear features but some nonlinearity would be ignored inevitably, and others (e.g., RF, DNN) focused on extracting nonlinear features for classification. All models are adopted in PC3 cell line with the most promising ratio of positive to negative.

A. Deep Learning Results

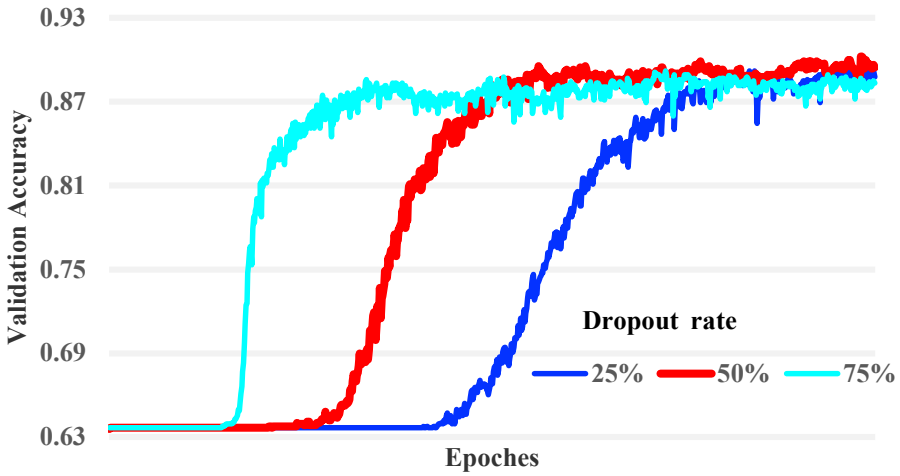
DNNs are flexible multilayer systems of connected and interacting artificial neurons that perform various data transformations. They have several hidden layers of neurons, which allowing the adjustment of the data abstraction level. The ability to learn at the higher levels of abstraction made DNNs a promising and effective tool for working with biological and chemical data. In the LINCS project, linear features can be captured by linear methods, but classification performance reaches an accuracy plateau, because such method does not capture complex nonlinear relationship between the expressions of genes. To systematically learn hierarchical nonlinear features, which are inevitably ignored by linear models, we designed a DNN as one input layer with 1,956 neurons corresponding to the dimensionality of features, two hidden layers with 200 and 10 neurons and one Softmax layer as binary classification layer. Therefore, each sample was presented by a 10-dimensional feature vector after feature extraction by DNN, and such feature learning effectively contributed to classification. However, overfitting is a serious problem in a fully connected network, and complicated network is time consuming when forward propagation. This is the reason why we adopted dropout, a technique for addressing this problem, to simplify network architecture. The key idea is to randomly drop hidden neurons during training to prevent hidden neurons from co-adapting too much [18]. Lastly, a trained model was used for discovering unknown DTIs. Overall performance of DNN with different architectures is shown in Table I. Furthermore, for the network to possess a better potential of prediction, the ratio of positive to negative was selected as 1:2 and the objective function was rebuilt by weighting all negative samples' probability of belonging to negative class, as defined in (7).

The whole process of framework, as shown in Fig. 2, contains feature fusion, negative data sampling, model training, and prediction. In the PC3 cell line, too many negative samples lead to serious class imbalance. Therefore, the input space consisted of all positive samples and a part of negative samples were uniform from the negative sample space. To model relationships between drugs and genes, these two data channels were used as a collection rather than separation. Therefore, before feeding into any model, each sample was created by fusion of a drug data and a gene data at feature level without any drops, because the original features, which were sufficient statistics, fully contain information of raw data at the feature level. Although direct methods to put drug data and gene data together through simple operation (e.g. addition, subtraction, multiplication, and division) were performed, these did not generate additional redundant features that led to a more complex feature space and risk of over-fitting. However, such operations were irreversible and the expressions of several key loci were changed, which resulted in the information lost. For the combination of original features from two data channels, we constructed an expression map for each sample in a serial manner. Such method preserved all original information and did not introduce more redundant noises. In the training procedure, the probability of dropout rate was selected as 50% (as shown in Fig. 3A), which means that the final model integrated 2^t sub-models, where t denotes the number of hidden neurons in second hidden layer, and the weight η in the objective function is selected as 10 according to observations of learning curve (as shown in Fig. 3B). Then, we used AdamOptimizer as an objective function optimizer with a learning rate of 1e−4 to make DNN fit training data with over 98% train accuracy, and to generalize validation data with approximately 90% validation accuracy, as shown in Fig. 5.

B. Ablation Study

We introduced different types of methods for ablation study, including LR, RF, Voting Classifier (VC), and Gradient Boosting Decision Tree (GBDT) on L1000 dataset

A.



B.

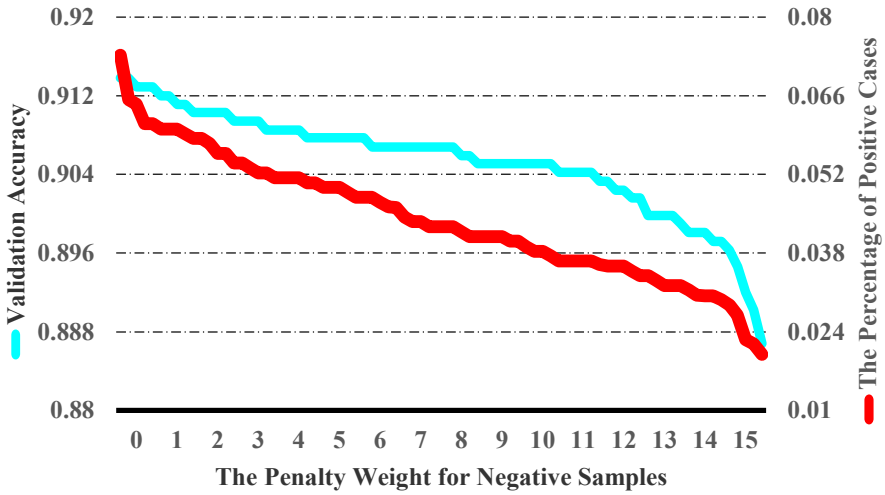


Figure 3. The experimental results of different hyper parameter settings: [A] The validation accuracy under different dropout rates. When the dropout rate equals to 50%, the performance of DNN is the best. Because the trained model is ensembled by 2^t sub-models, where t denotes the number of hidden neurons in second hidden layer. [B] The results under different penalty weights for negative samples. In order to keep tradeoff between validation accuracy and percentage of positive cases, the penalty weight for negative samples is selected as 10.

for evaluating the performance of DNN we designed by validation accuracy, F-score, proportion of positive cases and predictive error. LR is responsible for linear analysis because they are capable of capturing effective linear features. RF and VC, as ensemble classifiers consisting of multi weak classifiers, are widely adopted in classification tasks. GBDT showed amazing performance in the recommender system and potential purpose prediction owing to the advantage of combining different features.

As shown in Table II, validation accuracy and F-score of DNN are better than other methods, and the proportion of positive cases is at least six times lesser. In addition, if the predictive error, as defined in (8), of x_i is greater than 0, meaning that the performance of the model is worse than

TABLE II. PERFORMANCE COMPARISONS ACROSS METHODS

	Validation Accuracy	F-score	Percentage of Positive Cases
LR	76.83%±0.97	68.85%±0.06	38.07%±2.86
RF	87.13%±1.52	77.55%±4.99	23.43%±1.79
VC	90.03%±0.06	84.47%±0.72	29.89%±3.02
GBDT	90.45%±0.03	85.87%±1.71	28.40%±2.07
DNN	90.54%±1.45	86.39%±1.96	3.92%±1.06

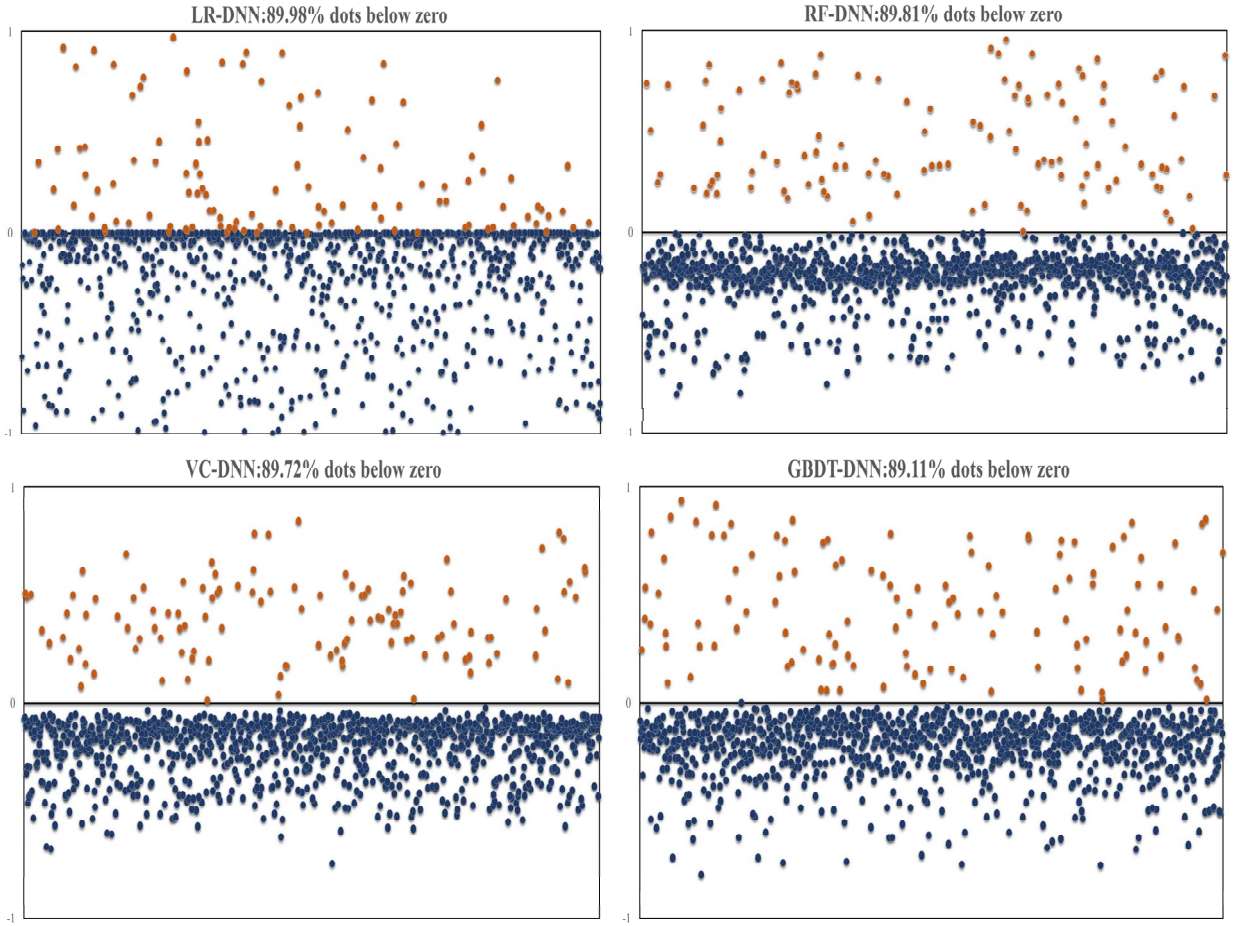


Figure 4. The predictive errors between DNN and other models. Each dot represents the difference between DNN and other models: If the dot has a negative label, the predictive error is from the $CM_{i,0}$ of other models minus $CM_{i,0}$ of DNN; otherwise, the predictive error is from the $CM_{i,1}$ of other models minus $CM_{i,1}$ of DNN. Therefore, if more dots are below the 0 horizontal line, the performance of DNN is better than other models; otherwise, the DNN performs worse.

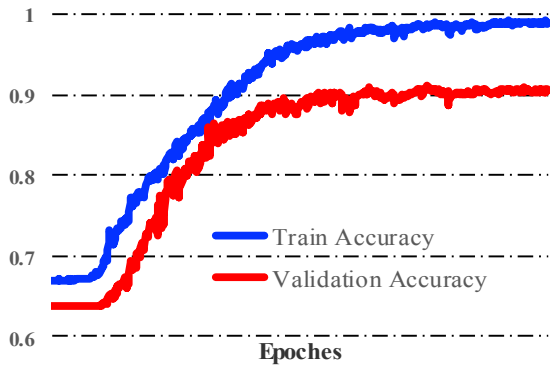


Figure 5. The performance of DNN under best hyper-parameter setting: The training accuracy is over 98% and the validation accuracy is about 90%.

DNN; otherwise, other models are shown to be better than DNN. As shown in Fig. 4, over 89% of the dots are below the 0 horizontal line in PC3 cell line. In other words, the results suggest that the performance of the DNN we designed is much better than other classical classification methods.

C. Validation of Predicted Results

We investigated the predicted interactions using other drug–target interaction databases, including TTD, MATADOR, IUPHAR/BPS, and STITCH. A total of 24 pairs are found in TTD, 185 pairs in MATADOR, 95 pairs in IUPHAR/BPS, and 221 pairs in STITCH. In addition, we ranked all predicted interactions by D-score, and computed the overlap pairs between the predicted results and interactions from other database. Then, we counted the number of overlap pairs in the sliding bins of 2,000 consecutive interactions (as shown in Fig. 6). Our framework can predict a certain part of novel DTIs validated by known experiments in other databases.

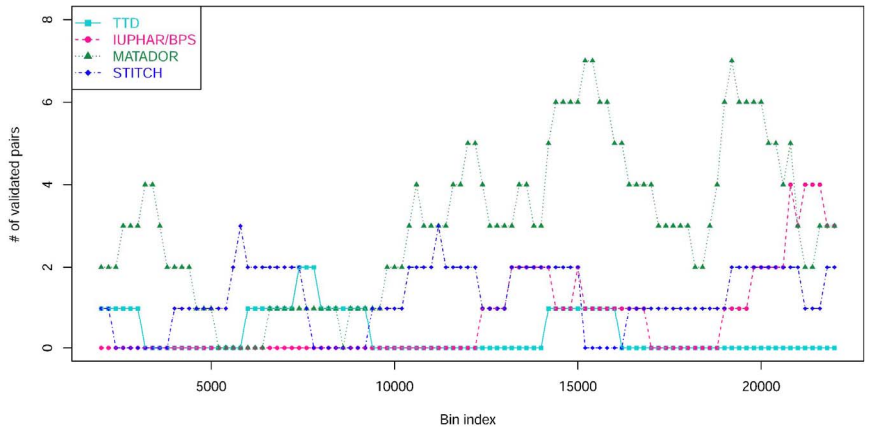


Figure 6. The overlap curves between predicted interactions and interactions validated by known experiments in other databases, including TTD, MATADOR, IUPHAR/BPS and STITCH: All the predicted interactions were ranked based on D-score. Furthermore, we calculated overlap interactions between predicted result and the interactions from the four database. The numbers of overlap interactions in the sliding bins of 2,000 consecutive interactions were counted.

IV. CONCLUSION

In this work, we proposed a framework to predict unknown DTIs based on transcriptome data of drug perturbation and gene knockout trails from the L1000 database. The whole pipeline of our framework includes combination of data from drugs and genes, negative data sampling, DNN training, and DTI prediction. Resulting from increasing availability of big data and GPU computing, the deep learning method in our framework worked as an effective tool for feature learning and classification. The results said our framework has ability to discovery more reliable DTIs and was further validated across platforms with high percentage of overlap interactions. That also suggested that our framework is capable of integrating transcriptome data from drugs and genes, and has potential and wider prospect for predicting DTIs for improving the drug discovery process.

ACKNOWLEDGMENT

This work was supported by National Nature Science Foundation of China [U1435222]; Program of International S & T Cooperation [2014DFB30020].

REFERENCES

[1] E. Lounkine, et al., Large-scale prediction and testing of drug activity on side-effect targets. Nature, 2011. 486(7403): p. 361-7.

[2] J. T. Dudley, T. Deshpande, and A. J. Butte, *Exploiting drug–disease relationships for computational drug repositioning.* Briefings in Bioinformatics, 2011. **12**(4): p. 303.

[3] V. Law, et al., DrugBank 4.0: shedding new light on drug metabolism. Nucleic Acids Research, 2014. 42(Database issue): p. 1091-7.

[4] S. Günther, et al., SuperTarget and Matador: resources for exploring drug-target relationships. Nucleic Acids Research, 2008. 36(Database issue): p. D919.

[5] X. Chen, Z. L. Ji, and Y. Z. Chen, TTD: Therapeutic Target Database. Nucleic Acids Research, 2002. 30(1): p. 412.

[6] M. Campillos, M. Kuhn, A. C. Gavin, L. J. Jensen, and P. Bork, Drug Target Identification Using Side-Effect Similarity. Science, 2008. 321(5886): p. 263.

[7] K. Bleakley and Y. Yamanishi, Supervised prediction of drug-target interactions using bipartite local models. Bioinformatics, 2009. 25(18): p. 2397-2403.

[8] Y. Wang and J. Zeng, Predicting drug-target interactions using restricted Boltzmann machines. Bioinformatics, 2013. 29(13): p. i126.

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

[10] Y. Yamanishi, M. Kotera, M. Kanehisa, and S. Goto, Drug-target interaction prediction from chemical, genomic and pharmacological data in an integrated framework. Bioinformatics, 2010. 26(12): p. i246.

[11] Z. Xia, L. Y. Wu, X. Zhou, and S. T. Wong, Semi-supervised drug-protein interaction prediction from heterogeneous biological spaces. Bmc Systems Biology, 2010. 4(S2): p. S6.

[12] Y. Chen, Y. Li, R. Narayan, A. Subramanian, and X. Xie, Gene expression inference with deep learning. Bioinformatics, 2016. 32(12): p. 1832.

[13] A. J. Pawson, et al., The IUPHAR/BPS Guide to PHARMACOLOGY: an expert-driven knowledgebase of drug targets and their ligands. Nucleic Acids Research, 2014. 42(D1): p. 1098-106.

[14] M. Kuhn, et al., STITCH 3: zooming in on protein-chemical interactions. Nucleic Acids Research, 2012. 40(Database issue): p. D876.

[15] M. Liu, R. Lang, and Y. Cao, Number of trees in random forest. Computer Engineering & Applications, 2015.

[16] A. Prinzie and D. V. D. Poel, Random Forests for multiclass classification: Random MultiNomial Logit ☆. Expert Systems with Applications, 2008. 34(3): p. 1721-1732.

[17] M. Zhang and H. Li, Tree kernel-based SVM with structured syntactic knowledge for BTG-based phrase reordering. in Conference on Empirical Methods in Natural Language Processing: Volume. 2009.

[18] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, Dropout: a simple way to prevent neural networks from overfitting. Journal of Machine Learning Research, 2014. 15(1): p. 1929-1958.