AI Winter is Coming

Mariya Popova[1], Stephen Capuzzi[1,2], Olexandr Isayev[1,*] (1) UNC Eshelman School of Pharmacy, University of North Carolina at Chapel Hill, NC 27516 (2) Currently at Vertex Pharmaceuticals, Boston, MA * Corresponding author; email: olexandr@olexandrisayev.com

Methods

To comply with challenge requirements about data/code availability, our final solution for Round 2 had to implement several shortcuts. We had to limit the dependence on commercial software/services as much as possible. Overall, this negatively affected most models vs. internal benchmarks. Therefore:

- Only RDKit fingerprints were used as a molecular representation. No other descriptors were used.
- Only one ML method was used - Gradient Boosted Decision Trees (GBDT) as implemented in XGBoost.
- Hyperparameter search was performed with a randomized search only. No custom GP or AutoML-like schemes were used.
- For simplicity, no model stacking was used.

Note, because of very large HPC compute demands our final prediction for MTOR, KDR, PLK1, SYK, TEK was done not with all 5 ensemble models, as they were not finished by the deadline. This Docker has all finished model and produces slightly different predictions for these kinases. This difference, however, does not affect any rankings.

Data

For all targets of interest, data was integrated from DrugTargetCommons and ChEMBL 25. Bioactivities were extracted for pChEMBL activity values -Log(IC50/EC50/Ki/Kd) of 10 μM or better, with ChEMBL CONFIDENCE_SCORE of 6 or greater for ‘binding’ or ‘functional’ human kinase protein assays. Due to conflicting naming schemes, all target datasets were integrated by Uniprot IDs. Each target dataset was curated according to our well-established best practices [J. Chem. Inf. Model.2010, 50 (7), 1189-1204 (https://doi.org/10.1021/ci100176x)]. Structural standardization, the cleaning of salts, and the removal of mixtures, inorganics, and organometallics was performed using ChemAxon software. In the case of replicate compounds, InChI Keys were generated. For replicates with the same activities in a given assay, a single representative compound was selected for inclusion into the training set. For replicates with the different activities (> 1 log unit) in a given assay, all compounds were excluded.

Feature Representation

An ensemble of four RDKit fingerprints of path length 5,7,9,11; each of 4K bit length. The total length of the feature vector was 16K.

Model Training.

All models were trained within two nested five-fold cross-validation loops. All splits were random. Internal CV loop was used to perform hyperparameters search and variable selection for the corresponding fold using the following protocol:

1. Fast XGBoost hyperparameter search with a budget of 100. The following parameters were tuned: max_depth, subsample, colsample_bytree, learning_rate, min_child_weight, gamma. n_estimators was fixed at 1000. For XGBoost we used fast histogram method on GPU and optimize for “log cosh” loss in order to reduce the effect of outliers.
2. Using the optimal model we performed recursive feature elimination as implemented in Scikit-Learn (RFECV function). A number of features that give model with the best MAE were selected.

3. Rebuild XGBoost model with the optimal number of features and perform an additional hyperparameter search with a budget of 500. The following parameters were tuned: n_estimators, max_depth, subsample, colsample_bytree, colsample_bylevel, learning_rate, min_child_weight, gamma, reg_alpha, reg_lambda. For XGBoost we used fast histogram method on GPU and optimize for “log cosh” loss in order to reduce the effect of outliers.

Final scoring was done by averaging five predictions from the external CV loop. Negative model bias was adjusted by scaling prediction values to the range of [5, prediction max]

Conclusion

This is the winning solution for RMSE metric.

Running the Docker Container

In order to run this Docker container, first pull the image using:

```
docker pull docker.synapse.org/syn18701196:9686282
```

Then, from a local directory containing an "input" and "output" directory, where "input" contains the file "input.csv" (the round_2_template.csv from IDG-DREAM or similarly formatted file), run the following command:

```
docker run -it --rm -v ${PWD}/input:/input -v ${PWD}/output:/output
docker.synapse.org/syn18701196:9686282
```

Deep and Shallow Chemogenomic Modelling for Compound-Target Binding Affinity Prediction Using Pairwise Input Neural Networks & Random Forests

Heval Ataş[1], Ahmet Rifaioglu[2], Tunca Doğan[1,3], Maria Jesus Martin[3], Rengul Atalay[1], Volkan Atalay[1,2]

[1] KanSiL, Department of Health Informatics Graduate School of Informatics, METU, Ankara, 06800 Turkey [2] Department of Computer Engineering, METU, Ankara, 06800 Turkey [3] European Molecular Biology Laboratory European Bioinformatics Institute (EMBL-EBI), Hinxton, Cambridge, CB10 1SD UK

Abstract

Machine learning techniques are frequently used in the field of drug discovery and repurposing for the prediction of interactions between drug candidate compounds and target proteins since the experimental approaches are not time- and cost-efficient to be applied to the massive compound-target interaction space. Recently, chemogenomic modelling approach became popular, where both compound and target protein features are used as inputs of the predictive models. Hence, they are able to incorporate targets with low number of (or no) training data and yield accurate predictions even for targets/compounds not involved in the training set at all. Chemogenomic approach is significant as it can be used to predict novel ligands for targets with limited training data and to identify the druggability potential of human proteins that were never targeted before. In this study, we developed chemogenomics-based computational methods, using random forest and deep neural network supervised learning techniques, to predict the binding affinities of a large set of kinases against several drug candidate compounds.

Introduction

The identification of binding affinity values between compounds and target proteins is critical for early stage drug discovery. Traditionally, binding affinity values are determined by high-throughput screening experiments, which are time-consuming and expensive, and thus, cannot be applied to the massive compound-target space. Therefore, computational methods have been developed to predict binding affinities, using machine learning (ML) techniques. Recently, chemogenomic modelling approaches became popular, where both compound and target protein features are used as the pairwise inputs of the predictive models. The output of these models are the binding affinity value predictions for the corresponding compound-target pair. The two main advantages of chemogenomic modeling are: (i) ability to incorporate targets with low number of (or none) training data points, (ii) potential to achieve elevated predictive performance due to more complex modeling. Here, we developed chemogenomics-based methods, on which, we tested different sets of compound and target protein features as input to observe the predictive performance.

Methods

In this study, we generated several predictive models (only 4 of them are shown here) for the binding affinity prediction of compound-kinase interactions using random forest (RF) and feed-forward pairwise input neural network (PINN) algorithms with different combinations of feature types and modeling approaches, as shown in Table 1. Unlike many conventional ML-based compound-target interaction prediction studies, where the prediction is usually based on binary classification as active or inactive, we generated regression models to predict the quantitative binding affinity values of compound-kinase interactions.

We represented compounds with ECFP4 fingerprints (diameter: 2), which is one of the most widely used feature type for compounds, and we represented proteins as pssm-based feature vectors (i.e., tri-gram-PSSM and k-separated-bigram-PSSM). POSSUM web-server was employed to generate the feature vectors. We obtained experimental bioactivity data points for kinases from the ChEMBL database for training, where we included all bioactivities containing a pChEMBL value (i.e., -log(IC50, EC50, Ki, Kd, Potency, …)). For our first model, we used the all kinase

interaction data points with 192,935 data points to train a single model. For our second model, we generated seven sub-models. Each sub-model was trained with the data points of a specific kinase sub-family such as: Agc, Camk, Cmgc, Ste, Tk, Tkl and others including 15,706, 13,251, 21,498, 4,165, 66,385, 10,470 and 29,982 data points, respectively. The aim here was to observe if family specific modeling increases the predictive performance. For Model 1 and 2, we used RF algorithm with tree number = 100 and max_features = 0.33. RF model takes a concatenated feature vector (compound + target) as input. The final part is a regressor, which predicts binding affinity for the input compound-target pair in terms of pChEMBL values.

For Model 3 and 4, we used pairwise input feed-forward neural networks (PINN) as a deep-chemogenomic neural network architecture. The network takes a pair of feature vectors for compounds and targets from disjoint input nodes simultaneously, following certain number of processing layers, latent representation of compound and target features are concatenated and further processed on more feed-forward layers. The output layer is a single node (a regressor), which predicts binding affinity for the input compound-target pair in terms of pChEMBL values. We used two hidden layers for both and compound target side of the network. After the concatenation of compound and target hidden layers, two additional hidden layers were used before output. We examined different hyper-parameters concerning the number of neurons at each layer (4096, 2048, 1536, 1024, 512, 128), learning rate (0.01, 0.001, 0.005, 0.0001) and dropout rate (0.6, 0.8) before finalizing the model.

We evaluated model performances by 5-fold CV and by external validation on the IDG DTI prediction challenge test dataset (i.e., the experimentally identified bioactivity measures between a selected set of kinases and compounds, these data point has not been recorded in any bioactivity databases such as ChEMBL or PubChem yet) using root mean squared error (RMSE), Pearson and Spearman correlations, and F1-score. For F1-score, the problem should be transformed to classification, for this, we determined an active/inactive predicted binding affinity threshold of pChEMBL = 7.

Results & Conclusion

Cross-validation results are given in Table 2. For Model 2, we reported weighted means of seven sub-models for each metric. Model performance comparisons are given below:

Model 1 vs. 2: the family-specific model outperformed the all-kinases model, which is an important outcome in terms of data selection and modeling approach. It is probable that the models trained with a more focused dataset (i.e., data points belong to the members of a kinase family) performs better, because different kinase families have different ligand interaction properties, and the model that contain all kinases at once cannot generalize the data at hand successfully.

Model 3 vs. 4: these models performed similarly, which indicates that the effect of the target feature type was minimal between k-sep-bigrams and trigrams features, which are similar in terms of the underlying representation logic, but very different in terms of dimensionality (k-sep-bigrams: 400, trigrams: 8,000). It is important to note that, we previously examined several more target feature types, and k-sep-bigrams and trigrams were selected based on those preliminary tests.

Model 1 vs. 3: RF models outperformed PINN models considering the cross validation results. As stated in the literature, the performance of deep neural network models are highly dependent on the selected hyper-parameters. Until this point, we could not scan a vast hyper-parameter space yet, we believe this is the main reason behind the observed performance difference

Information on IDG-DREAM Drug-Kinase Binding Prediction Challenge - Round2 Submission

RF Model (ObjectId: 9686327) For this submission we employed the methodology used in Model 1, as explained above; however, we reduced the training dataset to only the data points of the target kinases that are presented in the Round2 test dataset. The finalized training set was composed of 94,184 activity measurements between 61,603 compounds and 204 kinases.

PINN Model (ObjectId: 9686326) For this submission we employed the exact same methodology used in Model 3, as explained above.

Commands To Run the Docker Containers

RF Model (ObjectId: 9686327)

```
sudo docker run -it --rm -v $PWD:/input -v $PWD:/output
docker.synapse.org/syn18636383/crossbar_chemogenomic-modelling_rf:9686327
```

PINN Model (ObjectId: 9686326)

```
sudo docker run pinn-kinase-prediction-model
```

Authors Contribution Statement

HA, ASR, MJM, RA, VA and TD conceived the idea. HA and ASR performed the system construction. HA, ASR and TD performed all data analyses. MJM, RA, VA and TD supervised the overall study. All authors have revised and approved the this document.

Tables

Table 1: Characteristics of RF and PINN models.

Algorithm	Training set of each model	Protein Feature	Drug Feature
Model 1	RF	all kinases (1 model)	k-sep-bigrams
Model 2	RF	kinase families (7 sub-models)	k-sep-bigrams
Model 3	PINN	all kinases (1 model)	k-sep-bigrams
Model 4	PINN	all kinases (1 model)	trigram

Table 2. Predictive model performance results in 5-fold cross-validation.

Model name	RMSE	Pearson correlation	Spearman correlation	F1-score
Model 1 (RF)	0.64	0.87	0.87	0.85
Model 2 (RF)	0.63	0.87	0.87	0.86
Model 3 (PINN)	0.73	0.72	0.65	0.65
Model 4 (PINN)	0.73	0.72	0.64	0.65

Prediction of kinase inhibitor Kd values

Ádám Misák, Bence Szalai, László Hunyady, Gábor Turu Semmelweis University, Department of Physiology, Budapest, Hungary

Abstract/Summary

This prediction uses a stacked prediction of multiple learners to predict the Kd values of kinase inhibitors. As training data combined values of Ki and Kd data from Drug Target Commons (DTC) has been used, and features consisted of calculated molecular features (Morgan fingerprints, farmacophore features, autoencoder features and Tanimoto similarities to a selected set of kinase inhibitors) protein features (kinase sequence distance and ligand-inhibition correlation between kinases) and measured or inputed ligand displacement and kinase inhibition data. The resulting prediction scored 0.477 spearman correlation score on the Challenge dataset.

Methods

For data preparation, kinase data has been extracted from DTC database, Kd, Ki values have been converted to pKi and pKd values and averaged, and used together as ‘affinity’ data. Inhibition and activity data has been also extracted, combined together as remaining activity and used later as features. Additional data has been extracted from a publication (Drewry DH et al, Plos One, 2017), which was used as displacement feature in the final prediction. Although these two features are not readily available in a common ligand screening setup, since 2/3 of the challenge data had such displacement values for the specific protein-ligand pairs, we included these as features. For protein-compound pairs, where displacement and/or inhibition data were not available, the features were imputed using XGB regressor model. Molecular features were calculated with either rdkit library (Morgan features, 1024 bit, radius = 3, pharmacophore distance features, using the default pharmacophore descriptors and 16 bins and Tanimoto distances to the kinase inhibitor set used in previous publication (Drewry DH et al, 2017) using Morgan features), or pretrained neural network, built using Keras library (Gómez-Bombarelli, 2018, https://github.com/HIPS/molecule-autoencoder) (autoencoder features). As protein features, we used kinase domain sequence based distance map (distance matrix) and kinase to kinase correlation calculated from data from Drewry DH et al, 2017 (kinase correlation). Multiple models have been built with XGB regressor (https://xgboost.readthedocs.io/en/latest/), catboost (https://github.com/catboost/catboost), LGBM regressor (https://lightgbm.readthedocs.io) and scikit-learn’s Ridge regressor (https://scikit-learn.org/). The regressors trained and stacked with an XGB regression model using vecstack (https://github.com/vecxoz/vecstack).

Conclusion

The final model scored well on spearman corralation and AUG scores in the challenge, but the rmse scores were not satisfactory.

References

Drewry DH, Wells CI, Andrews DM, Angell R, Al-Ali H, Axtman AD, Capuzzi SJ,Elkins JM, Ettmayer P, Frederiksen M, Gileadi O, Gray N, Hooper A, Knapp S, Laufer S, Luecking U, Michaelides M, Müller S, Muratov E, Denny RA, SaikatenduKS, Treiber DK, Zuercher WJ, Willson TM. Progress towards a public chemogenomic set for protein kinases and a call for contributions. PLoS One. 2017 Aug 2;12(8):e0181585.

Gómez-Bombarelli R, Wei JN, Duvenaud D, Hernández-Lobato JM, Sánchez-Lengeling B, Sheberla D, Aguilera-Iparraguirre J, Hirzel TD, Adams RP, Aspuru-Guzik A. Automatic Chemical Design Using a Data-Driven Continuous Representation of Molecules. ACS Cent Sci. 2018 Feb 28;4(2):268-276. doi: 10.1021/acscentsci.7b00572

Team Prospectors: Ensemble based semi-supervised approach to IDG-DREAM Drug-Kinase Binding Prediction Challenge

Davor Oršolić, Bono Lučić, Višnja Stepanić, Tomislav Šmuc Ruđer Bošković Institute, Zagreb, Croatia

Background

We have provided submissions for both Round 1 and Round 2 based on IDG-DREAM Drug-Kinase Binding Prediction Challenge Data: 14,492 drugs and 1,462 proteins. Both rounds were addressed by ensemble based models (Random Forest and Boosting trees), with the main effort put into feature set construction, instance selection and model selection. Final Round 2 model has been produced using *XGBoost* algorithm and feature set based on selected similarities in compound and protein space [6].

Methods

In the exposition of the methodology used for this challenge we refer to the original training set as to the compound-target interactions with valid, known Kd values, available for all challenge participants via Synapse challenge portal. The term training set is also used when referring to particular subsets of this original training set used for model optimization/selection, or developing specific models for the submissions in Round 2. We use term validation set which typically means test set obtained by training/validation set partitioning from the original training set in order to be able to test and select/optimize models. When referring to test set we usually refer to Round 1 or Round 2 test set – used for scoring submissions.

Data preprocessing & Feature engineering

For the prediction of drug-target interactions of given pairs we first retrieved protein FASTA sequences and SMILES representations of compounds from UniProt and ChemSpider, respectively. Some effort has been spent on data cleansing and preprocessing. Feature set describing compounds and targets for the initial round of modeling (Round 1) were constructed using *rcdk* and *protr* R packages, respectively [4][7]. We experimented with different subsets of compound features, but found out that maccs fingerprints gave best results. For testing and optimization of the modeling workflow we split training set of drug-kinase interactions into 70/30 ratio. In later stages (Round 1 and Round 2) we completely changed the feature set and used similarities in compounds and targets space in order to describe drug-target pair. To estimate similarities in target space, EMBOSS program with *needleall* application was used to globally align all pairs of primary protein structures [8]. To align protein sequences by the EMBOSS program, Needleman-Wunsch algorithm was used, and the EBLOSUM62 matrix for calculation of similarities, together with other default settings. Compound similarities were based on precalculated *maccs* fingerprints from R's rcdk package depending on *kekule* SMILES representation [4]. To determine similarities between all compounds, *fingerprint* package was utilized and Tanimoto coefficients were calculated based on comparison of 166 maccs keys for each of 14492 compounds [5].

Modeling

Round 1

In our Round 1b experiments we investigated different feature sets and used random training/validation splitting (70/30) of the original dataset of compound-target interaction pairs in order to improve performance of the predictive models. We have tested different types of compound descriptors, including similarities. First submitted models in

Round 1b were based on maccs fingerprints (166 features) and protein physical descriptors (41 features). We had also introduced similarity based approach in few of our Round 1b models which showed improvement in comparison with our first submission.

Round 2

In Round 2 modeling we focused more on similarity based representation of the problem. We also introduced new training/validation set definition in order to use more representative (similar) instances with respect to the interactions from the Round 2 scoring test set. For that purpose we first clustered joint training set with Round 2 test set - using *hclust* algorithm from stats package in R - into 100, 200, 400 clusters (using target and compound similarity matrices) [1]. We than formed reduced training/validation set from only those compound-target interactions that had compounds clustered together with test set compounds and targets clustered together with test set targets. We used clustering results to redefine similarity feature sets, too. Similarity feature sets were based on cluster representatives from compound space and cluster representatives from the target space, which were used as „anchors“ for similarity features on which we regressed instances from the training set. Using the optimized *XGBoost* scheme we tested models using following compound + target similarity feature sets (100+100, 200+200, 400+400)[6]. The training set with 200+200 similarity feature sets and 13,786 compound-target interactions was used to train the models for the first Round 2 submission.

Final submission - Compound-target selection for the training set

Final submission for the Round 2 and the best result was the *XGBoost* model which was trained on the training set based on compound-target interactions for which targets are one of the test targets or the targets that were clustered together with some of the test targets. This subset of interactions was further filtered to only those that have compounds clustered together with test set compounds.

Final submission - Feature construction

For the final submission we used the „anchors“ for similarity features, a subset of test targets, as well as targets from target-clusters containing test set targets which are not available in the training set. Similarly compound similarity feature set was based on the partitioning of the compound space into clusters. For compound "anchors" for similarity we used most similar compounds from clusters containing compounds from the test set (with avg.similarity>0.5). This meant that our final model was trained using 7,336 compound-interaction pairs and 207 + 194 similarities as features, from compound and target similarity matrices respectively.

Algorithms and model selection/optimization

During the course of the two rounds of the challenge we experimented with two ensemble based algorithms: we started using Random Forest (*randomForest* R package) and Round 1 submissions were based on the models produced using RF [3]. The models were based on 500 and 2000 trees, respectively, controlling for the depth/complexity of the trees by limiting the size of terminal nodes to 80 samples. For the Round 2 we used Boosting trees – or *XGBoost* algorithm implementation of R package [6]. We optimized the algorithm parameters using *train* function from *caret* package [9].

The tuning parameter grid had the following parameters: *max_depth* (maximum tree depth, default: 6) *eta* (learning rate) *gamma* (used for regularization tuning) *colsample_bytree* (column sampling, default: 1) *subsample* (row sampling, default: 1) *min_child_weight* (minimum leaf weight, default:1) This parameter optimization was performed on the reduced training set (6,450 pairs) and using 3-fold cross validation.

Discussion

We unfortunately entered the challenge very late, and have managed to produce first models on time before Round 1b was closed. This models were based on simple set of features describing compounds and targets, and large portion of

the original training set was used for the model development. Our results in Round 1 were of low quality (Spearman correlation < 0.1; AUC ~ 0.55; RMSE ~ 1.1). In Round 2 we started to experiment more with compound/target similarities as features upon which regression ensemble models were based. We also used clustering of training/test interaction pairs that served several purposes: (i) as the way to extract more meaningful training and validation set for the model development; (ii) try to focus our model on the instances in the neighborhood of test set instances; (iii) use clusters as means for feature selection, as we used similarity matrices in compound/target space as features to make regression models. Our final Round 2 submission results were (Spearman correlation=0.296; AUC=0.685; RMSE=1.196) which represented significant improvement from the Round 1 results. Our findings from the Round 2 experiments show that the approach based on similarities is promising approach for the treatment of this type of the problem (large and diverse set of compounds and targets), and that learning methodology should be capable to capture highly non-linear and very localized interactions – in that respect learning models based on smaller number of samples in close proximity of test samples (learning in the neighborhood – or proximity of the actual tested interaction pairs) is better than learning from large, non-localized training set.

Authors Statement

DO implemented the code and performed calculations. TS, BL, VS, conceived and supervised the study. TS and DO wrote the text (write-up).

Command required to run the docker container:

```
docker run -it --rm -v ${PWD}/io:/input -v ${PWD}/io:/output
docker.synapse.org/syn18553372/prospectors_idg:9686257
```

References

[1] R Core Team (2013). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. ISBN 3-900051-07-0, URL http://www.R-project.org/. [2] Breiman, L. (2001). "Random forests." Machine learning 45.1: 5 32 [3] Liaw, A. and Wiener, M. (2002). Classification and Regression by randomForest. R News 2(3), 18--22. [4] Guha, R. (2007). 'Chemical Informatics Functionality in R'. Journal of Statistical Software 6(18) [5] Guha R. (2006). fingerprint: Functions to Operate on Binary Fingerprint Data. R package version 2.2, URL http://CRAN.R-project.org/. [6] Chen, T. and Guestrin, C. (2016). XGBoost: A Scalable Tree Boosting System. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '16). ACM, New York, NY, USA, 785-794. DOI: https://doi.org/10.1145/2939672.2939785 [7] Nan Xiao, Dong-Sheng Cao, Min-Feng Zhu, and Qing-Song Xu. (2015). protr/ProtrWeb: R package and web server for generating various numerical representation schemes of protein sequences. Bioinformatics 31 (11), 1857-1859. [8] Rice, P., Longden, I. and Bleasby, A. (2000). EMBOSS: The European Molecular Biology Open Software Suite. Trends in Genetics. 16(6):276-277 [9] Kuhn, M. (2008). Caret package. Journal of Statistical Software, 28(5)

DreamChallenge Report - DruginaseLearning Team

Christos Fotis[1], Panagiotis Terzopoulos[1], Konstantinos Ntagiantas[1] and Leonidas G.Alexopoulos[1,2]

1. BioSys Lab, National Technical University of Athens, Athens, Greece.

2. ProtATonce Ltd, Athens, Greece.

Introduction

Our efforts focused on building a deep end-to-end model similar to Ozturk et al. (Ozkirimli Hakime Ozturk and Arzucan Ozgur. Deepdta: Deep drug-target binding affinity prediction.arXiv:1801.10193, 2018.), that takes as input the SMILES representation of a compound and the amino acid sequence of a protein and outputs the KD value of the compound-protein pair. On this front, we investigated several deep architectures in order to represent the SMILES-sequence pairs in a latent space that best captures the nature of the binding affinity prediction. Furthermore, a considerable amount of our effort focused on augmenting the initial DTC dataset, with more compound-protein pairs having available KD values in the literature.

Methods

Data and augmentation

The initial dataset from DTC was augmented using various compound-kinase binding datasets that are publicly available in the web and in the literature. Overall, compound-kinase pairs with KD values from DTC, BindingDB, KKB, PKIS, HMS LINCS and Davis et al. were combined to create the final dataset for training and validation. The final dataset consisted of over 105K unique drug-protein interactions labeled with the Kd affinity metric. A detailed report of this work can be found on https://github.com/bsl-ntua.

Models

We experimented with different end-to-end architectures that utilize different methods for the latent representation of the SMILES and a deep CNN for the latent representation of the amino acid sequences.

1. The first architecture used a 3 layer deep graph convolutional network similar to (Jorge Aguilera-Iparraguirre Rafael Gomez-Bombarelli Timothy Hirzel AlanAspuru-Guzik David Duvenaudy, Dougal Maclauriny and Ryan P. Adams. Convolutional networks on graphs for learning molecular fingerprints. NeuralInformation Processing Systems (NIPS), 2015.), to extract application specific neural fingerprints from the compound structures. These fingerprints were then concatenated with the output of a 3 layer deep CNN that encodes the amino acid sequences of the proteins. The combined feature vector was fed through 2 fully connected layers for the final KD prediction. Batch normalization layers and relu activations were used throughout the network except for the final prediction layer. In order to reduce overfitting, dropout and L2 regularization was used between the fully connected layers.
2. Regarding the second architecture, a deep LSTM autoencoder was first trained on the SMILES sequences of all the compounds in the training, validation and test sets. The autoencoder used as input the one-hot representation of the SMILES and was tasked to predict the next letter in the sequence. The output of the trained encoder can serve as a compressed latent representation of the SMILES space. The idea behind training an autoencoder first, is that the encoder learns to represent all the available SMILES (training and test) and the final model should

perform better than a fully end-to-end architecture that has never seen the structures of the test set. Thus, the final model used as input the output of the encoder along with the one-hot encoded amino acid sequences. The sequences were encoded again using a 3 layer deep CNN and concatenated with the output of the encoder to build the final feature vector. This feature vector was then passed through 2 fully connected layers for the final KD prediction. Batch normalization layers and relu activations were used throughout the network except the final prediction layer. In order to reduce overfitting, dropout and L2 regularization was used between the fully connected layers.

Training and Evaluation

The final augmented dataset consisted of 105431 unique pkd values between 12041 compounds and 1690 kinases, with more than 70000 pairs having a pkd value close to 5 (KD=10000μM). In order to reduce the bias of the trained model towards inactive compounds we decided to filter the interactions resulting in a final 3:1 ratio between inactive (*pkd*<7) and active pairs (*pkd*>=7). For model evaluation and parameter tuning a competition specific 5-fold cross validation scheme was employed. More specifically, we identified 5 sets of compounds, with similarity profiles with the training set, almost identical to the similarity profiles of the test set. During each step of the cross-validation all interactions that included the compounds of the validation set were used for model evaluation and the rest for model training. The data augmentation pipeline was implemented using R while the models were built in python using keras with tensorflow as back end. Training was performed on a NVIDIA GPU GTX-1080Ti.

Results and discussion

The best predictions for the test set came out of the second architecture we implemented which included the encoder. Having in mind how difficult it is to really generalize to new compound scaffolds never previously seen during training (Izhar Wallach and Abraham Heifets. Most ligand-based classification bench-marks reward memorization rather than generalization. J. Chem. Inf.Model., 58:916–932, 2018.), an encoder that has been trained to represent the combined train and test-set distribution is expected to boost performance when its encoded feature vector is fed for further training.
As a disclaimer we should say that the predictions we submitted in the final round of the competition are not the true ones our model predicts. This is because of a code error we discovered, unfortunately, after the submission deadline, which implicitly changes the dictionary according to which the SMILES are encoded to 1hot arrays every time one loads the model. This means that the autoenoder was trained with a different 1hot encoding than the one used for the test compounds predictions. We strongly believe that if it wasn't for this error, our latest submission would have scored substantially better.

Docker instructions

The command to run the docker is:

```
$ docker run -it --rm -v ${PWD}:/input -v ${PWD}:/output
docker.synapse.org/syn18525357/druginase-model:9686322
```

As per the instructions on the contest's website, the script included in the docker container reads a provided file with the name "input.csv" that is similar to the round 2 template file, as well as the .h5 files necessary to load our pre-trained keras deep-learning model to make the predictions. All of these files are put into the /input directory of the docker container. Running the docker locally with the provided command, requires the input file and the two .h5 files in the working directory, and will result in the creation of the "predictions.csv" file, that is stored in the /output directory of the docker container. In the instructions it is stated that the provided file should be named "input.csv" or "template.csv", and we decided to go with the former. In order for the container to run succesfully, an active Internet connection is required on the host machine, as the script queries the website of UniProt, in order to retrieve the protein

sequences of the respective Uniprot_IDs of the kinases provided in the "input.csv" file. In a system with a moderately stable Internet connection, this may take up to 15 minutes. If the querying is successful, a message will be printed with the number of proteins queried equal to the number of interactions in the "input.csv" file. It should be noted that to run the container locally, not only the "input.csv" file is necessary, but also the two .h5 files that contain the trained model, namely "docker_model22_4.h5", and "test_encoder21.h5".

Authors contribution statement

C. Fotis and K. Ntagiantas conceived a significant part of the presented idea, did most of the data augmentation work in R and helped with fitting the model and interpreting the results. P. Terzopoulos wrote most of the python code and custom functions needed to feed the data through the keras pipeline and trained the models. L.G. Alexopoulos supervised the project.

Protein ligand binding prediction by neural networks and gradient boosted trees ensemble

Mehmet Tan TOBB University of Economics and Technology Dept. of Computer Engineering Ankara, Turkey

Abstract

This document explains the basic methodology followed for our submission in the IDG-DREAM Drug-Kinase Binding Prediction Challenge. A total of 394 kinase binding predictions were required in this challenge. Binding data only from Drug Target Commons database is exploited. However, three other databases were used for the representation of chemicals and proteins. Our submission has achieved a final RMSE score of 1.113 in round 2.

Introduction

Protein ligand binding score prediction is a fundamental issue in drug discovery. It has a great potential to decrease the large cost of the drug discovery process. In this DREAM challenge, the binding scores of the protein-chemical pairs were to be predicted by exploiting available knowledge in binding databases.

The first database to be used is the Drug Target Commons (DTC) database [1]. It has a large number of protein ligand binding data in terms of different metrics such as the IC50, Ki and Kd. DTC is the only database we used for binding data. In addition to this database, we used the EBI-ChEMBL [2] for protein and chemical representations. We used the basic representations of SMILES and amino acid sequences for chemicals and proteins respectively.

The following sections describes the methodology and conclusions.

Methods

Data sets

The challenge requires the prediction in terms of pKd. Therefore, as a first step, we extracted the rows corresponding to the Kd values from the DTC database. The other types of binding data did not improve the results for our model so we did not include them in our dataset. This amounts to 55678 protein ligand binding data samples.

For chemicals, we used the well-known Extended Connectivity FingerPrints (ECFP) as the descriptors. We downloaded the SMILES strings for each of the compounds from the EBI-ChEMBL database and used the rdkit ibrary [3] to produce 512 length binary bit vectors to represent the compounds. Similarly, for representing kinases, we downloaded the amino acid sequences from EBI-ChEMBL in fasta format. From those sequences, we used PyBioMed library [4] to construct descriptors of proteins. From this library, we used the composition, transition and distribution features (calculateCTD method from CTD module) and dipeptide composition features (CalculateDipeptideComposition method from the AAComposition module) as the feature set. We simply concatenated these two feature vectors to build a descriptor for proteins.

As a second preprocessing step, we removed those features whose variance are below a threshold. For this, we used the VarianceThreshold module of the scikit-learn library [5] with a threshold of 0.1 for chemicals and proteins separately. After this, we obtained descriptors vectors of length 185 and 206 for chemicals and proteins respectively. Finally, we standardised the feature vectors separately to zero mean and unit standard deviation. Therefore, at the end, our final descriptor for a compound-protein pair is a vector of length 391, a simple concatenation of the above mentioned vectors.

Models

The machine learning models that we use to model the protein ligand binding data is neural networks and gradient boosted decision tree models. These two models are the best performing models from a number of other models that we executed.

The neural network model we use is a network of two hidden layers of size 1500 and 400 respectively. In addition to that an input layer of 391 and output layer of a single neuron exist in the network. The activation functions of hidden layers are sigmoid functions and the output neuron has a linear activation function. We exploited the Keras library [6] to construct and train this network. We trained the network with a batch size of 128 for 400 epochs by using ADAM method for the optimization.

For gradient boosted trees (GBT), we used the official implementation [7] of the efficient LightGBM algorithm [8]. This algorithm usually works faster than other GBT methods by exploiting the size of gradients for the samples in the dataset and eliminating those that have a small sized gradient. We used all the default values for the parameters except that the number of estimators is set to 3960 which is the value we found by parameter optimization.

After training these two models, the final predictions are produced by a linear ensemble of these. The ensemble is a simple weighted model that computes $$(w_1 * o_{nn} + w_2 * o_{gbt})$$ where $$(o_{nn})$$ and $$(o_{gbt})$$ are the predictions of neural network and GBT model for a given test sample, respectively. Based on a simple parameter optimization, we observed that an equal weighting gives the best results, therefore we set $$(w_1 = w_2 = 0.5)$$.

Conclusion

This submission has used one type of binding data, the Kd values. In the future, we plan to extend the work here to also exploit the other types of binding data as well. Also the performance can be improved by extending the ensemble with other models.

Author Contribution Statement

This is a single author submission.

Running the docker image

Please run the following command to run the image associated:

```
docker run -it --rm -v ${PWD}/io:/input -v ${PWD}/io:/output demo
```

where 'demo' represents the name of the image and 'io' is a folder in the host machine for the input (the template.csv file in the challenge).

References

[1]Tang et al. Drug Target Commons: A Community Effort to Build a Consensus Knowledge Base for Drug-Target Interactions. Cell Chem Biol. 2018 Feb 15;25(2):224-229.e2

[2] Gaulton A, Hersey A, Nowotka M, Bento AP, Chambers J, Mendez D, Mutowo P, Atkinson F, Bellis LJ, Cibrián-Uhalte E, Davies M, Dedman N, Karlsson A, Magariños MP, Overington JP, Papadatos G, Smit I, Leach AR. (2017) 'The ChEMBL database in 2017.' Nucleic Acids Res., 45(D1) D945-D954.

[3] RDKit: Open-Source Cheminformatics Software. (https://www.rdkit.org)

[4] PyBioMed Molecular descriptors library. (https://github.com/gadsbyfly/PyBioMed)

[5] Scikit-learn: Machine Learning in Python, Pedregosa et al., JMLR 12, pp. 2825-2830, 2011.

[6] Chollet, François. Keras. (2015) (http://keras.io)

[7] LightGBM Library. (https://github.com/Microsoft/LightGBM)

[8] Ke, Guolin, et al. "Lightgbm: A highly efficient gradient boosting decision tree." Advances in Neural Information Processing Systems. 2017.

Predicting compound-kinase binding interactions using convolutional neural networks by combining SMILES and aligned ATP binding sites

AmsterdamUMC-KU-team Team: Georgi K. Kanev[1], Albert J. Kooistra[2], Bart A. Westerman[1]

[1] Department of Neurosurgery, Cancer Center Amsterdam CCA, De Boelelaan 1117, 1081 HZ Amsterdam, The Netherlands [2] Department of Drug Design and Pharmacology, University of Copenhagen, Universitetsparken 2, 2100 Copenhagen, Denmark

Introduction

Convolutional neural networks (CNNs) represent the current state-of-the-art algorithms in image and video recognition. [1] In the recent years, CNNs were applied to predict bioactivity,[2-3,9] learn molecular fingerprints,[4] detect chemical motifs,[5] and predict properties of small molecules[6]. In this work we used SMILES (Simplified Molecular Input Line Entry System) and sequences of aligned ATP binding sites (85 amino acids) of protein kinases downloaded from the KLIFS database containing[7-8] as input for the CNNs to predict the compound-kinase binding interactions. SMILES are widely used for encoding molecular structures and represent compounds in the form of a string over a fixed set of characters, describing all the atoms and structure of small molecules including chirality, bonds, aromaticity and more. The KLIFS (Kinase-Ligand Interaction Fingerprints and Structures)[7-8] database systematically aligns and process all current human and mouse protein kinase structures, focusing on the interactions of ligands in the binding site of protein kinases, assessment of binding pockets, kinase motifs and overall kinase and ligand properties. The representation of the compounds (SMILES string) and protein kinases (aligned ATP binding sites) in the form of a 2D matrix, allows CNNs to identify import motifs and map them to compound-kinase binding interactions.

Methods

Prediction

The data from ChEMBL (v24.1), DrugTargetCommons, IUPHAR/BPS Guide to pharmacology, and literature was integrated and curated. The Kd, Ki and IC50 measurements in combination with a protein kinase were filtered out and used for training of the CNN. The integrated data set comprised of 298,595 compound-kinase measurements, 439 unique kinases and 101,189 compounds. The SMILES of the compounds and the sequences of the ATP binding sites (downloaded from KLIFS) were one-hot-encoded and used as input for the 2D convolutional layers of CNN (Figure 1). The CNN comprised of a single 2D convolutional and 2D max pooling layer for the SMILES (shape=33,156,1) and single 2D convolutional and 2D max pooling layer for the sequences (shape=21,85,1). The convolutional layers used 64 filters. After both max pooling layers, dropout (0.5) was applied. The output of the dropouts was given to dense layers, each with 256 nodes. In addition to these layers, 2 dense layers were used - one received ECFP-4 fingerprints as input and the other received kinase family as input (one-hot encoded). By complementing the one-hot encoding of SMILES with a chemical fingerprint such as the ECFP-4, the chemical information can probably be better encoded into features for the CNN. The output of all described layers was concatenated and given to a dense layer with 32 nodes. A dropout of 0.4 was applied and the output was given to the output layer.

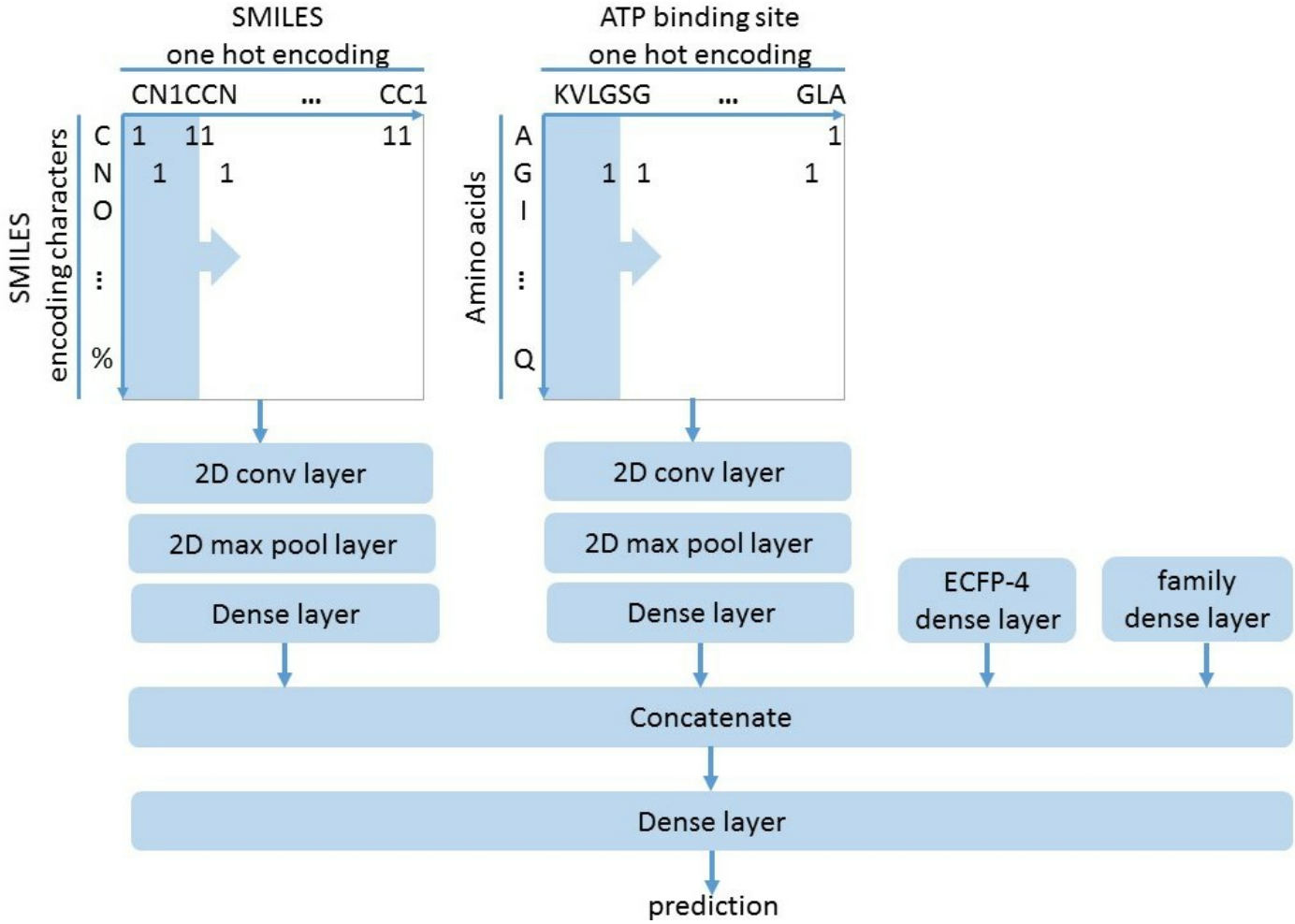


Figure 1. The convolutional neural network used to predict the compound-kinase binding interactions.

Refinement

After applying the model the predicted pKd values were further refined using literature data from both DTC/ChEMBL and the KiEO (Kinase Experiments Omnibus (http://tanlab.ucdenver.edu/KIEO/KIEOv1.0/). This simplistic refinement approach was applied as follows for each kinase-compound pair:

1. If one or more fully characterized (p)Kd/(p)Ki/(p)IC50 values were available from literature the median value was used instead of the prediction.
2. If step 1 was not applied , then all (p)Kd/(p)Ki/(p)IC50 values for highly similar compounds for the same kinase target (Tanimoto score 0.6 using the Morgan fingerprint RDKit) were collected and the minimum, maximum and average were calculated. If no data for similar compounds was available, then the following step was applied. If the difference between the maximum and the minimum value was smaller than 0.1, the minimum and maximum values were changed to the average value - 0.5 and + 0.5, respectively to account for potential larger variations in the data. Subsequently, if the predicted value was outside the range of the current minimum and maximum value, the predicted value changed to the average value.
3. If none of the previous steps were applied and a minimum expected pKd value was available (from the single concentration KINOMEscan for the PKIS2 dataset [10]), then the predicted value was increased with 0.5 if the predicted value was below the minimum expected pKd value.
4. If none of the previous steps were applied and the predicted pKd value was higher than 6 the prediction was scaled up: predpKD + (predpKD-6)*0.5. This step performed, as we noticed that the predictions for which literature data was available (step 1) the predicted value was overall lower than the literature values.

Finally, to correct for outliers values above > 9.5 were scaled down to 9.5. As no values below 5 were present, not lower limit was applied.

The literature data applied in steps 1-3 is available here: https://www.synapse.org/#!Synapse:syn18635344

Conclusion

Here we show that by using an unbiased approach to train a CNN network with 2D convolutional layers, we are able to predict the bioactivity of compounds with a RMSE of 1.125 and a rounded average AUC of 0.658. Despite this, the low spearman correlation (0.259) indicates that the model was probably overfitted and requires further optimization and validation.

References

[1] LeCun et al. "Deep learning", Nature volume 521, pages 436–444 (2015) [2] Wallach et al. "AtomNet: A Deep Convolutional Neural Network for Bioactivity Prediction in Structure-based Drug Discovery", arXiv preprint arXiv:1510.02855 (2015) [3] Jiménez et al. "KDEEP: Protein–Ligand Absolute Binding Affinity Prediction via 3D-Convolutional Neural Networks", Journal of chemical information and modeling 58.2 (2018): 287-296 [4] Duvenaud et al. "Convolutional Networks on Graphs for Learning Molecular Fingerprints", Advances in Neural Information Processing Systems , 2015, pp. 2224–2232 [5] Hirohara et al. "Convolutional neural network based on SMILES representation of compounds for detecting chemical motif", BMC Bioinformatics. 2018; 19: 526 [6] Goh et al. "Using Rule-Based Labels for Weak Supervised Learning: A ChemNet for Transferable Chemical Property Prediction", Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. ACM, 2018 [7] van Linden et al. "KLIFS: A Knowledge-Based Structural Database To Navigate Kinase–Ligand Interaction Space", Journal of Medicinal Chemistry, 2014, 57 (2), pp 249–277 [8] Kooistra et al. "KLIFS: a structural kinase-ligand interaction database", Nucleic Acids Research 44.D1 (2015): D365-D371 [9] Stepniewska-Dziubinska et al. "Development and evaluation of a deep learning model for protein-ligand binding affinity prediction", Bioinformatics. 2018;34(21):3666-74. [10] Drewry et al. "Progress towards a public chemogenomic set for protein kinases and a call for contributions", PloS one. 2017; 12(8):p.e0181585.

Authors Contribution Statement

GKK, data integration, data analysis and development models; AJK, data integration, data analysis and prediction refinement; BAW, project initiation and supervision

Title: Q.E.D's solution to IDG-DREAM Drug-Kinase Binding Prediction Challenge

Authors

Fangping Wan[1, 4] (Username: Stig, Email: wfp15@mails.tsinghua.edu.cn), Shuya Li[1, 4] (Username: Shuya Li, Email: lishuya17@mails.tsinghua.edu.cn), Yunan Luo [2](Username: Yunan Luo, Email: yunan@illinois.com), Hailin Hu[3, 4] (Username: Corgi, Email: huhl16@mails.tsinghua.edu.cn), Jian Peng[2] (Email: jianpeng@illinois.edu), Jianyang Zeng[1, 4] (Email: zengjy321@tsinghua.edu.cn)

1: Institute for Interdisciplinary Information Sciences, Tsinghua University, Beijing, China.
2: Department of Computer Science, University of Illinois at Urbana-Champaign, Urbana, USA.
3: School of Medicine, Tsinghua University, Beijing, China.
4: Silexon Co., LTD, Beijing, China.

1. Summary

This project contains the submission 2 (objectID 9686285) method from Q.E.D team, for both sub-challenges of IDG-DREAM Drug-Kinase Binding Prediction Challenge round2.

2 .Methods - submission 2 (objectID 9686285)

2.1 Data pre-processing

Generally, we used the compound-protein affinity data from Drug Target Commons (DTC) (Tang J, Ravikumar B, Alam Z, et al. DrugTargetCommons:a community effort to build a consensus knowledgebase for drug-target interactions. Cell Chemical Biology, 2018, 25(2):224-229.e2.) (downloaded from https://www.synapse.org/#!Synapse:syn17017461).

2.1.1 Used compounds:

Among the compounds curated in DTC, we only considered compounds that (1) have CHEMBL ID or (2) have PKD affinity. Compounds that have CHEMBL ID have structure information encoded by InChI or SMILES (stored in DTC). Compounds that have PKD affinity are encoded as InChIkey in DTC. We used PubChem Identifier Exchange Service (https://pubchem.ncbi.nlm.nih.gov/idexchange/idexchange.cgi) to map InChIkey to corresponding SMILES (stored as "DTC_pkd_inchikey_to_smiles" in the data folder in docker).

2.1.2 Used proteins:

Among the proteins curated in DTC, we only considered proteins that (1) have Uniprot ID and were labeled as kinase in Uniprot or (2) came from round_2_template.csv (i.e., test compounds, downloaded from https://www.synapse.org/#!Synapse:syn16809885).

2.1.3 Used affinities:

Among the binding affinity pairs curated in DTC, we only considered the pairs that satisfied the following conditions:

(1) compounds that came from 2.1.1; (2) proteins that came from 2.1.2; (3) the binding affinity measurement type was Ki or KI or Kd or KD or EC50 or PKD; (4) the binding affinity measurement relation was "=" (i.e., equal); (5) if the binding affinity measurement type was Ki or KI or Kd or KD or EC50, the standard unit should be "NM"; (6) In DTC dataset, each row represents a binding affinity, the "target id" column (i.e., the fifth column) should contain only one Uniprot ID, otherwise (e.g., multiple Uniprot IDs) we did not use this row.

We used the binding affinity pairs that satisfied the above conditions. Then, for the pairs that had the type Ki or KI or Kd or KD or EC50, we converted the binding affinity x using the transformation: $-\log_{10}(x / 10^9)$. For pairs that had the type PKD, we did not do such transformation. Some compound-protein pairs can have multiple binding affinities. In such cases, the median of the binding affinities was used. Note that, the median operation was applied after the $-\log_{10}(x / 10^9)$ transformation.

After the above pre-processing, the total number of compounds we used in training process is 13,608; the total number of proteins we used in training process is 527; and the total number of binding affinities we used in training process is 60,462.

2.2 Prediction method

Under the problem setting that requires fine-grained discrimination between similar compounds or targets, we found the explicit introduction of similarity metrics as model input generally outperformed other more complex model architectures that attempt to organize representative features from scratch. In particular, we defined a comprehensive set of compound structure similarity and protein sequence similarity metrics as the input of our model. Then, we leveraged CGKronRLS method (Pahikkala T. Fast gradient computation for learning with tensor product kernels and sparse training labels[C]//Joint IAPR International Workshops on Statistical Techniques in Pattern Recognition (SPR) and Structural and Syntactic Pattern Recognition (SSPR). Springer, Berlin, Heidelberg, 2014: 123-132.) (implemented in Pahikkala T, Airola A. RLScore: regularized least-squares learners[J]. The Journal of Machine Learning Research, 2016, 17(1): 7803-7807. https://github.com/aatapa/RLScore) as the regression model to predict the binding affinity.

2.2.1 Features of compounds

We computed the following compound similarity matrices as compound features (computed by RDKit: https://github.com/rdkit/rdkit):

1: Tanimoto similarity of morgan fingerprint with arguments radius=2, nBits=1024, useChirality=True.

2: Tanimoto similarity of morgan fingerprint with arguments radius=2, nBits=1024, useChirality=False.

3: Tanimoto similarity of morgan fingerprint with arguments radius=3, nBits=1024, useChirality=True.

4: Tanimoto similarity of morgan fingerprint with arguments radius=3, nBits=1024, useChirality=False.

5: Dice similarity of morgan fingerprint with arguments radius=2, nBits=1024, useChirality=True.

6: Dice similarity of morgan fingerprint with arguments radius=2, nBits=1024, useChirality=False.

7: Dice similarity of morgan fingerprint with arguments radius=3, nBits=1024, useChirality=True.

8: Dice similarity of morgan fingerprint with arguments radius=3, nBits=1024, useChirality=False.

2.2.2 Features of proteins

We computed the protein similarity matrix as protein features (computed by https://github.com/mengyao/Complete-Striped-Smith-Waterman-Library). Specifically, protein similarity is defined as the normalized Striped-Smith-Waterman similarity. Let sw(s1, s2) be the alignment score of Striped-Smith-Waterman algorithm between protein sequences s1 and s2. The protein similarity between s1 and s2 can be defined as sw(s1, s2) / sqrt(sw(s1, s1) * sw(s2, s2)).

2.2.3 Regression model

We used CGKronRLS as the regression model. It took the compound and protein similarity matrices as input and output the binding affinity.

2.2.4 Model ensemble

Instead of using single model, we used the ensemble of multiple CGKronRLS (with different iterations, regularization parameters and input features) models. We ensembled 440 CGKronRLS models with the following setting: protein feature ∈ {the protein similarity matrix from 2.2.2} x compound feature ∈ {eight compound similarity matrices from 2.2.1} x regularization parameter of CGKronRLS ∈ {0.1, 0.5, 1.0, 1.5, 2.0} x iteration of CGKronRLS ∈ {400, 410, 420, 430, 440 ,450, 460, 470, 480, 490, 500}, where x means cartesian product.

After training the 440 CGKronRLS models, we averaged the predictions among them to produce the final prediction.

3. Testing environment

Submission 2 (objectID 9686285) model was trained and tested on a server with the following configuration: System version: Ubuntu 16.04.2 LTS; Cores: Intel(R) Xeon(R) CPU E5-2630 v3 @ 2.40GHz, 32 in total; Memory: 264024700 kB.

4. Running the final model

To run our docker image and get only the final output file for our submission 2 (objectID 9686285), run the following command:

```
$ docker run -it --rm -v ${PWD}/input:/input -v ${PWD}/output:/output
docker.synapse.org/syn18519352/qed-sub2:9686285
```

To run our docker image and get all the output files of intermediate processes for our submission 2 (objectID 9686285), run the following command:

```
$ docker run -it --rm -v ${PWD}/input:/input -v ${PWD}/output:/output -v
${PWD}/data:/data -v ${PWD}/SW_based_prediction:/SW_based_prediction
docker.synapse.org/syn18519352/qed-sub2:9686285
```

5. Author contribution

F.W., S.L., Y.L., H.H., J.P., and J.Z. conceived the method. F.W. and S.L. conducted the data pre-processing. F.W., S.L., Y.L. and H.H. calculated the features. F.W. and S.L. wrote and ran the regression model. S.L. prepared the docker with the help of F.W. F.W. wrote the writeup with support from all authors.

IDG-DREAM Drug-Kinase Binding Prediction Challenge ThinNguyen Writeup

Thin Nguyen
Applied Artificial Intelligence Institute, Deakin University, Australia
thin.nguyen@deakin.edu.au

In this challenge given a training drug-target affinity matrix, the task is to estimate the empty cells in the matrix. We evaluate several approaches on benchmark databases of similar problems where the ground-truth is available and suggest to use the best on the validation datasets, deep learning models, to tackle the challenge.

1 Introduction

Protein kinases are enzymes that catalyze phosphorylation reactions within the cells, thus regulating cell function. More than 500 kinases have been identified, representing approximately 2% of the human genome [6]. About 30% of human proteins may be modified by kinase activity [8], making kinases attractive targets for drug interventions. Measuring drug-kinase interactions through clinical trials is costly and time-consuming [3, 7]. Estimating the strength of the interactions for novel couples of drug-kinase based on the interactions already measured becomes an important alternative, where the challenge is a crowd effort.

Apparently the challenge could be considered as a collaborative filtering problem (CF). For example, in movie ratings as in the Nexflix competition[1], the rating for a couple of movie-user is learned, or collaboratively filtered, from the ratings by the movies/users similar to the given movie/user. The lesson from Nexflix competition is that if the number of training user-movie ratings is big enough, external information for users or movies does not make significantly contribution to the recommendation systems. However this is not always the case for drug-target binding prediction problem, where the affinity available is often sparse.

Another approach is kernel based, as in [2, 1]. In these work, kernels for drugs and targets are built from their molecular descriptors, input into a regularized least squares regression model (RLS) to predict the binding affinity.

For the challenge, the information of drugs, which are novel, is limited, making it difficult to compute biologically sensible kernels or similarity matrices among drugs, and hence the performance of CF or kernel-based methods could be compromised. It

[1] https://www.netflixprize.com/rules.html

is worse, or even inapplicable, when the drugs/proteins in the couples to be predicted the affinity are not in the training sets.

On the other hand, deep learning based modeling becomes a suitable approach when only 1D representation for drugs (SMILES) and proteins (sequences) is provided. Then the model learned from training data can predict the affinity for couples of drugs-targets in testing data with their 1D representation provided, regardless they are in the training data or not.

2 Methods

2.1 Collaborative filtering (CF)

For drug-target binding prediction problem, from the training affinity matrix we could build similarity matrices for both drugs and targets. Any distance measure can be used to calculate the similarity, e.g., cosine or correlation. With these similarity matrices at hand we can run drug-based or target-based filtering to estimate the binding power for unknown drug-target couples.

For example, for drug-based collaborative filtering, for an input couple (*d,t*), its affinity is estimated as the weighted sum of all other drug's affinity for target *t* where the weighting is the cosine similarity between *d* and other drugs. Similarly, for target-based collaborative filtering, for the same input couple (*d,t*), the affinity for the couple is estimated as the weighted sum of all other targets' affinity for drug *d* where the weighting is the cosine similarity between *t* and other targets.

$$\hat{a_{dt}} = \frac{\sum_{d'} sim(d, d') a_{d't}}{\sum_{d'} sim(d, d')}$$

$$\hat{a_{dt}} = \frac{\sum_{t'} sim(t, t') a_{dt'}}{\sum_{t'} sim(t, t')}$$

To void noise, instead of all, top K most similar drugs or targets can be used to estimate the affinity.

Another problem is that the scale of affinity differ among drugs and targets, i.e., some drugs have extreme low or high affinity for all targets. So a relative difference from the average can be used instead of the absolute affinity.

$$\hat{a_{dt}} = \bar{a_d} + \frac{\sum_{d'} sim(d, d') (a_{d't} - \bar{a_{d'}})}{\sum_{d'} sim(d, d')}$$

$$\hat{a_{dt}} = \bar{a_t} + \frac{\sum_{t'} sim(t, t') (a_{dt'} - \bar{a_{t'}})}{\sum_{t'} sim(t, t')}$$

where $\bar{a_d}$ and $\bar{a_t}$ is the average affinity for drug *d* and target *t*, respectively.

drug_id	target_id	avg_global	avg_drug	avg_target	affinity	SDR1	SDR2	SDR3	SDR4	SDR5	STG1	STG2	STG3	STG4	STG5
3	157	5.451527	5.166657	6.170107	5.0	5.000000	5.000000	5.000000	6.0	5.0	5.000000	5.0	5.000000	5.000000	5.00000
40	417	5.451527	5.881638	5.673105	5.0	5.000000	5.508638	5.000000	5.0	5.0	6.408935	5.0	5.000000	5.000000	5.00000
26	244	5.451527	5.265768	5.639389	5.0	5.000000	6.136677	5.000000	5.0	5.0	5.698970	5.0	5.259637	5.161151	5.00000
36	292	5.451527	5.716675	5.257152	5.0	8.004365	6.119186	9.244125	5.0	5.0	5.000000	5.0	5.000000	5.508638	5.79588
4	317	5.451527	5.247893	5.196341	5.0	5.000000	5.000000	5.000000	5.0	5.0	5.000000	5.0	5.000000	5.000000	5.00000

Figure 1: Represent a couple of drug-target through its neighbors in both drugs and targets.

Joint similarity collaborating filtering (joint-sim CF) While the above CF models suggest use either drugs or targets similarity to recommend affinity for a novel couple of drug-target, we suggest to use both for the task. In particular, given a couple of drug-target needs evaluating the affinity, we can represent it through the affinity scored by its *K* neighbors, in both drugs-based and targets-based. An example is shown in Figure 1, where *K*=5: the couple of (*drug_id*, *target_id*) is represented by *SDR1*,...,*SDR5* – the affinity with *target_id* scored by *K* drugs closest to *drug_id* and *STG1*,...,*STG5* – the affinity with *drug_id* scored by *K* targets closest to *target_id*.

2.2 Kernel based (KronRLS)

Given the problem is to predict the affinity for *n* drugs and *m* targets, there would be *n*m* combinations of them and the kernel would be in the size of $(n*m)^2$. To speed up model training, Cichonska et al. [2, 1] suggest to use KronRLS (Kronecker regularized least-squares). In KronRLS, a pairwise kernel *K* is computed as the Kronecker product of compound kernel of size *n*n* and protein kernel of size *m*m*.

2.3 Deep learning

2.3.1 Auto-encoder (DL_AE)

Auto-encoder based modeling is claimed to be state-of-the-art model for Netflix dataset[2]. In this approach, the training affinity matrix is assumed to be fully filled and is the input, and output, for an auto-encoder, as shown in Figure 2. Then the loss function is adjusted for ignoring not-in-the-train cells.

2.3.2 Embedded nodes in bipartite graph (DL_bipartite)

We consider the training affinity matrix as a bipartite graph and learn a continuous feature representation for drugs and targets by node2vec [5] for the graph, as illustrated in Figure 3a. Then the node vectors are concatenated in a neural network to predict the affinity for testing data, as shown in Figure 3b.

[2]https://paperswithcode.com/sota/collaborative-filtering-on-netflix

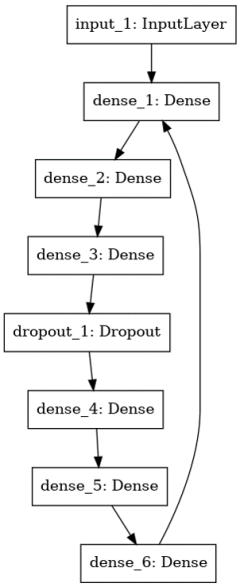


Figure 2: Auto-encoder of drug-target affinity matrix.

2.3.3 External information: SMILES and sequences (DL_1D)

No external information is used in the two deep learning models DL_AE and DL_bipartite. However, in the testing data provided by the organizer, for drugs, SMILES strings is given, and for targets, protein sequences can be retrieved with the UniProt_Id given [3]. These strings can be seen as 1D representation for drugs and proteins, input into a neural network to learn a model to predict the binding affinity for novel drug-kinase couples, as shown in Figure 4. In the figure, input_1 and input_2 are drugs and targets, respectively. As these are in 1D representation, layers of 1D convolutions and pooling are used to capture potential patterns in the inputs. They are then concatenated, sent through regularized layers of Dropout, and finally regressed with the training affinity.

3 Model validation

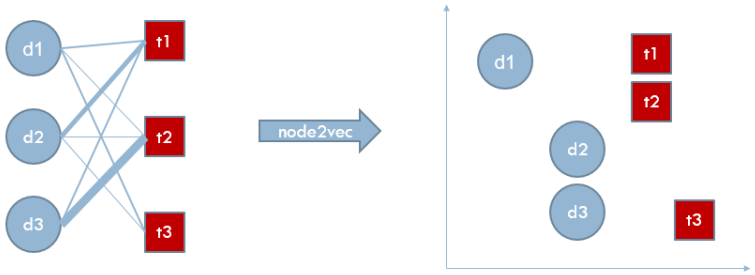
To seek a good model for the challenge we experimented the candidate models above with benchmark datasets of similar problems.

3.1 Datasets

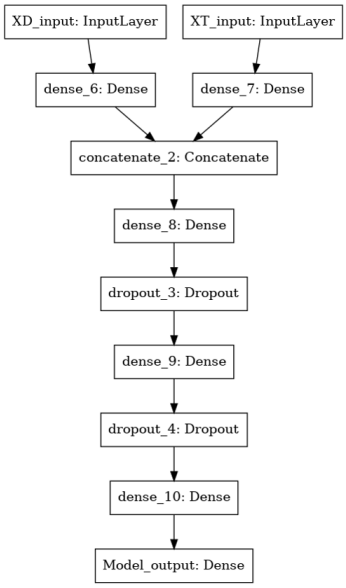
Two datasets were used to evaluate the models:

- **Davis** dataset: binding affinities observed for all pairs of 72 drugs and 442 targets, measured by Kd value (kinase dissociation constant) [4].

[3] https://www.uniprot.org/uniprot



(a) Learning a dense representation for drugs and targets, preserving their similarity in the bipartite graph.



(b) Drug and target vectors as input for building a model to predict the affinity.

Figure 3: Predicting binding affinity through the bipartite graph.

Model	rmse	Spearman corr
CF	1.28	0.46
DL_AE	0.86	0.27
joint-sim CF	0.69	0.57
KronRLS	0.58	**0.69**
DL_bipartite	0.56	0.65
DL_1D	**0.51**	0.68

(a) For Davis dataset. Best result is in bold.

Model (Kiba)	rmse	Spearman corr
CF	4.29	0.21
DL_AE	2	0.31
KronRLS	0.6	0.74
joint-sim CF	0.55	0.76
DL_bipartie	0.53	0.78
DL_1D	**0.43**	**0.85**

(b) For Kiba dataset. Best result is in bold.

Table 1: Prediction performance.

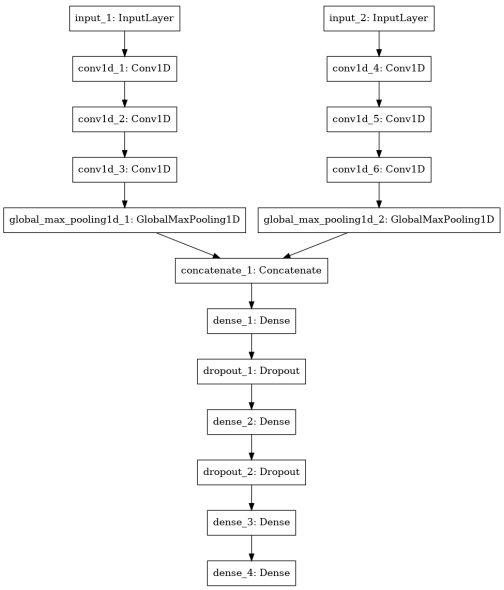


Figure 4: Predicting the affinity using external information.

- **Kiba** dataset: binding affinities for 2,116 drugs and 229 targets [9].

80% of data instances were used for training and 20% were for testing the models. Same data splitting was used for learning all the models.

3.2 Results

The result for all models mentioned above on the two datasets is presented in Table 1. **DL_1D** is best in two measures for Kiba dataset. It is also best in RMSE and second best in Spearman correlation for Davis dataset.

4 Model and data for the challenge

As **DL_1D** performs really well in the two benchmark datasets, we choose it as the model for the challenge.

Model is now trained on Drug Target Commons (DTC) data [10]. Only those tuples with the standard_type of ["KD","Kd","KD'", "PKD"] is selected. For those with ["KD","Kd","KD'"], the standard value is converted to pKd unit. 55,816 compound-protein pairs were included in this training data for 13,651 distinct compounds and 1,489 distinct proteins.

Model is validated on all Davis data [4]. There are 30,056 compound-protein pairs in this validation data.

The model gaining smallest RMSE for validation data is then used to predict the affinity for testing data.

5 Running model

- On GPUs:

docker login docker.synapse.org docker run --runtime=nvidia -it -v ${PWD}/io_gpu:/output docker.synapse.org/syn18518883/my-model:gpu

- On CPUs:

docker login docker.synapse.org docker run -it -v ${PWD}/io_cpu:/output docker.synapse.org/syn18518883/my-model:cpu

- Notes:

For running on GPUs, 'nvidia-docker' should be installed[4].

For our computers, training on GPUs is about 4.5 times faster than on CPUs, 4,003 seconds versus 18,078 seconds.

References

[1] Anna Cichonska, Tapio Pahikkala, Sandor Szedmak, Heli Julkunen, Antti Airola, Markus Heinonen, Tero Aittokallio, and Juho Rousu. Learning with multiple pairwise kernels for drug bioactivity prediction. *Bioinformatics*, 34(13):i509–i518, 2018.

[2] Anna Cichonska, Balaguru Ravikumar, Elina Parri, Sanna Timonen, Tapio Pahikkala, Antti Airola, Krister Wennerberg, Juho Rousu, and Tero Aittokallio. Computational-experimental approach to drug-target interaction mapping: A case study on kinase inhibitors. *PLoS Computational Biology*, 13(8):e1005678, 2017.

[3] Philip Cohen. Protein kinases–the major drug targets of the twenty-first century? *Nature Reviews Drug Discovery*, 1(4):309, 2002.

[4] Mindy I Davis, Jeremy P Hunt, Sanna Herrgard, Pietro Ciceri, Lisa M Wodicka, Gabriel Pallares, Michael Hocker, Daniel K Treiber, and Patrick P Zarrinkar. Comprehensive analysis of kinase inhibitor selectivity. *Nature Biotechnology*, 29(11):1046, 2011.

[5] Aditya Grover and Jure Leskovec. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 855–864. ACM, 2016.

[6] Gerard Manning, David B Whyte, Ricardo Martinez, Tony Hunter, and Sucha Sudarsanam. The protein kinase complement of the human genome. *Science*, 298(5600):1912–1934, 2002.

[7] Martin EM Noble, Jane A Endicott, and Louise N Johnson. Protein kinase inhibitors: Insights into drug design from structure. *Science*, 303(5665):1800–1805, 2004.

[4] https://github.com/NVIDIA/nvidia-docker

[8] Shawn J Stachel, John M Sanders, Darrell A Henze, Mike T Rudd, Hua-Poo Su, Yiwei Li, Kausik K Nanda, Melissa S Egbertson, Peter J Manley, Kristen LG Jones, et al. Maximizing diversity from a kinase screen: Identification of novel and selective pan-Trk inhibitors for chronic pain. *Journal of Medicinal Chemistry*, 57(13):5800–5816, 2014.

[9] Jing Tang, Agnieszka Szwajda, Sushil Shakyawar, Tao Xu, Petteri Hintsanen, Krister Wennerberg, and Tero Aittokallio. Making sense of large-scale kinase inhibitor bioactivity data sets: a comparative and integrative analysis. *Journal of Chemical Information and Modeling*, 54(3):735–743, 2014.

[10] Jing Tang, Zia ur Rehman Tanoli, Balaguru Ravikumar, Zaid Alam, Anni Rebane, Markus Vähä-Koskela, Gopal Peddinti, Arjan J. van Adrichem, Janica Wakkinen, Alok Jaiswal, Ella Karjalainen, Prson Gautam, Liye He, Elina Parri, Suleiman Khan, Abhishekh Gupta, Mehreen Ali, Laxman Yetukuri, Anna-Lena Gustavsson, Brinton Seashore-Ludlow, Anne Hersey, Andrew R. Leach, John P. Overington, Gretchen Repasky, Krister Wennerberg, and Tero Aittokallio. Drug target commons: A community effort to build a consensus knowledge base for drug-target interactions. *Cell Chemical Biology*, 25(2):224 – 229.e2, 2018.

Team:Boun - DeepDTA: Deep Drug Target Binding Affinity Prediction

Hakime Öztürk[1], Elif Ozkirimli[2], and Arzucan Özgür[1] 1 Department of Computer Engineering, Bogazici University, Istanbul, Turkey 2 Department of Chemical Engineering, Bogazici University, Istanbul, Turkey

Abstract

For the IDG-DREAM Drug-Kinase Binding Prediction Challenge, we adopted a deep-learning based approach named DeepDTA [1] that was previously introduced by our team. DeepDTA aims to predict interaction strenghts of protein-compound pairs by utilizing Convolutional-Neural Network (CNN) blocks. These blocks learn high-level representations of proteins and compounds from their respective sequences. We employed DeepDTA with default parameter settings to predict binding affinities of Kinase-drug interactions. The source code for DeepDTA is available here: https://github.com/hkmztrk/DeepDTA

Introduction

The IDG-DREAM Drug-Kinase Binding Prediction Challenge aims to address the task of predicting affinities of the interactions between Kinases and drugs in terms of K_d (dissociation constant) values . The challenge consisted of three Rounds, Round1, Round1b, and Round2. We participated in Round1 and Round2. In both rounds, we used the DeepDTA model as the prediction system. In Round1, we obtained our best performance with the public dataset Davis [2] as training dataset, whereas in Round2, we obtained our best performance with the filtered subset of Drug Target Commons (DTC) dataset. Here we discuss the results of our best performance in Round2.

Methods

Training and test data

We used Drug-Target Commons (DTC) as our training data set. We filtered the original dataset based on the activity types that are related to K_d (e.g., pK_d, log K_d) and converted them to pK_d, values. After the filtration, we obtained total 50181 interactions between 1353 proteins and 11902 targets. We will refer to this dataset as (DTC_filtered) from now on. As for test dataset, the dataset provided for Round2 was used. Table 1 summarizes the training and test datasets.

Data	#interactions	#proteins	#drugs
Training	50181	1353	11902
Round2 Test	394	25	207

DeepDTA requires the sequence information of proteins and drugs in order to perform prediction. For both datasets, SMILES information for the drugs were available. Whereas for proteins, we utilized Python Bioservices [3] to collect protein sequences from the UniProt [4] database using respective UniProt IDs of the proteins.

Prediction Model

In this challenge, we adopted DeepDTA model [1] which is a Convolutional Neural Network (CNN) based architecture to predict binding affinities of drug-target interactions. The model consists of two respective CNN blocks, each learning

high level representations from the sequences of proteins and drugs. The model combines these vectors into a single protein-drug representation. Finally, the concatenated representation is fed into a Fully-Connected Feed Forward Neural Network that consists of three layers with two dropout layers in between. Figure 1 below illustrates the architecture of the DeepDTA model.

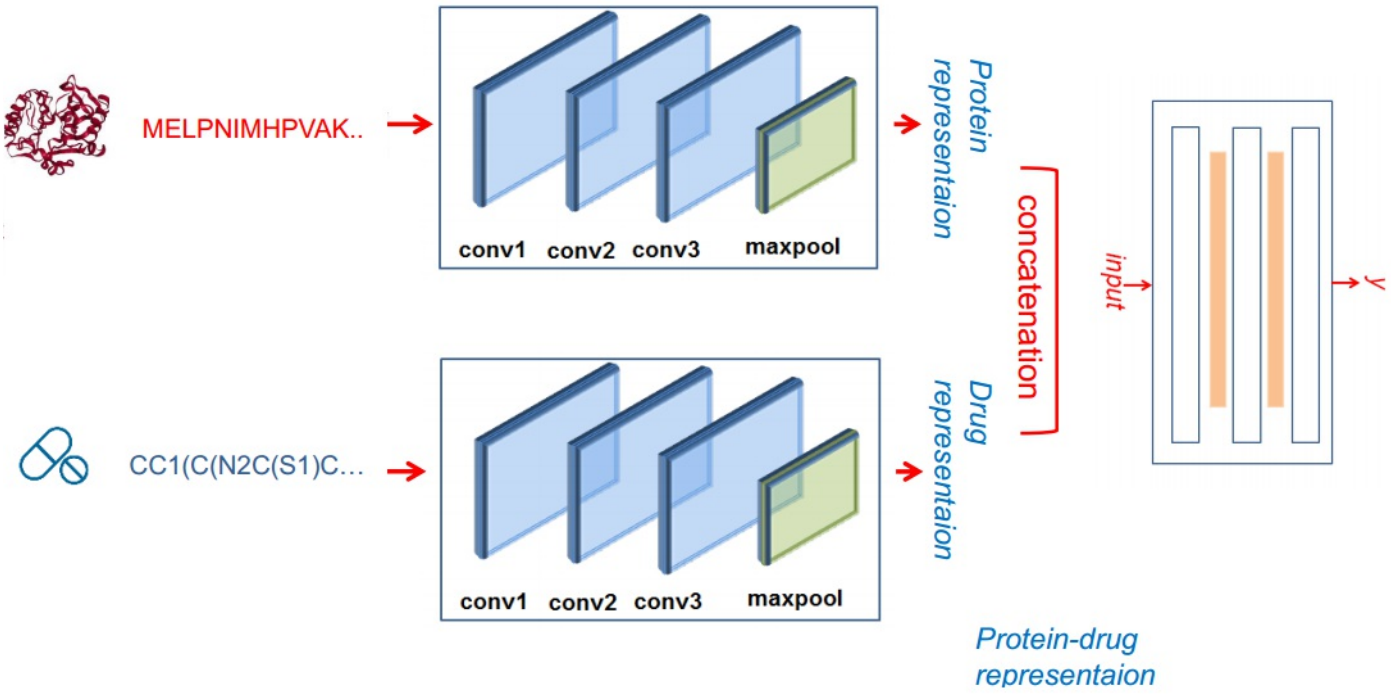


Figure 1: The summary of the DeepDTA architecture

We trained the model using the whole training data that we obtained after filtering (DTC_filtered) using the default parameter setting. Table 2 reports the default values of parameters that we utilized.

Parameter	value
Num. kernels	32, 64, 96
Protein kernel size	8
Drug kernel size	4
Batch size	256
Num. epoch	100
Dropout	0.1
FC	1024, 1024, 512

Keras [5] with Tensorflow [6] background is employed to build and train the DeepDTA model.

Conclusion

DeepDTA obtained values of 1.156 and 0.652 in terms of RMSE and AUC metrics in Round2 which scored average places in the Leaderboard. Since the model was trained with default values without any fine-tuning, the model

achieves a promising results using only sequence information of proteins and compounds.

References

[1] Öztürk, Hakime, Arzucan Özgür, and Elif Ozkirimli. "DeepDTA: deep drug–target binding affinity prediction." Bioinformatics 34.17 (2018): i821-i829.

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

[3] Cokelaer, Thomas, et al. "BioServices: a common Python package to access biological Web Services programmatically." Bioinformatics 29.24 (2013): 3241-3242.

[4] Apweiler R.et al. . (2004) Uniprot: the universal protein knowledgebase. Nucleic Acids Res. , 32(Suppl. 1), D115–D119.

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

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

Docker Instructions

Note: You have to place "input.csv" under the same directory as "Dockerfile"

```
docker build -t docker.synapse.org/syn18507647/deepdta:9686233 .

docker run -t -d -v  [your-dir]:/output
docker.synapse.org/syn18507647/deepdta:9686233

docker run -it --rm -d -v [your-dir]:/output -v  [your-dir]:/input
docker.synapse.org/syn18507647/deepdta:9686233
```

N121's solution to IDG-DREAM Drug-Kinase Binding Prediction Challenge

Chih-Han Huang 1,2,†, Edward S. C. Shih 1, †, Tsai-Min Chen 1, †, Chih-Hsun Wu1, Wei-Quan Fang1, Jhih-Yu Chen1, and Ming-Jing Hwang 1,2,*

1 Institute of Biomedical Sciences, Academia Sinica, Taipei, Taiwan, 2 Genome and Systems Biology Degree Program, National Taiwan University and Academia Sinica, Taipei, Taiwan *To whom correspondence should be addressed. †These authors contributed equally.

Abstract

Our model utilizes random forest for predicting the pKd by Circular Fingerprints of SMILES of drugs and inhibition data of proteins.

Introduction

Previous studies of prediction of pKd usually used structure or sequence data of protein. However, these methods cost lots of time to predict; furthermore, some of proteins do not have structure data. Experiments of measure pKd are hard, but to measure inhibition data of a protein and a drug is much more easier and faster. Here, we developed novel method to predict the pKd by Circular Fingerprints of SMILES of drugs and inhibition data of proteins.

Methods

Our model utilizes random forest for predicting the pKd by Circular Fingerprints of SMILES of drugs and inhibition data of proteinÂ˜s.

pKds , SMILES of drugs, proteins were obtained on Drug Target Commons [1]. Circular Fingerprints (Morgan Fingerprints) were achieved from SMILES of drugs by RDKit [2, 3]. Inhibition data of proteins was obtained from Jing Tang et al. and David H. Drewry et al. [1, 4]. Inhibition data of the specific protein and the specific drug and circular Fingerprint of the specific drug as feature for following training. Training set and testing set were randomly split as 70% and 30% Random forest was used to predicting the pKd using inhibition data of the specific protein and the specific drug and circular Fingerprint of the specific drug as feature. Training set was used for building model, and testing set was used for testing the model. AUC, F1, RMSE, pearson correlation coefficient, spearman correlation coefficient were used for evaluating the model performance.

Result

Performance of testing set was following: AUC 0.98, F1 0.92, RMSE 0.47, pearson correlation coefficient 0.95, spearman correlation coefficient 0.95. Fig. 1 was the plot of predicted pKd and reference pKd. Table 1 was result of round 1b, and Table 2 was result of round 2.

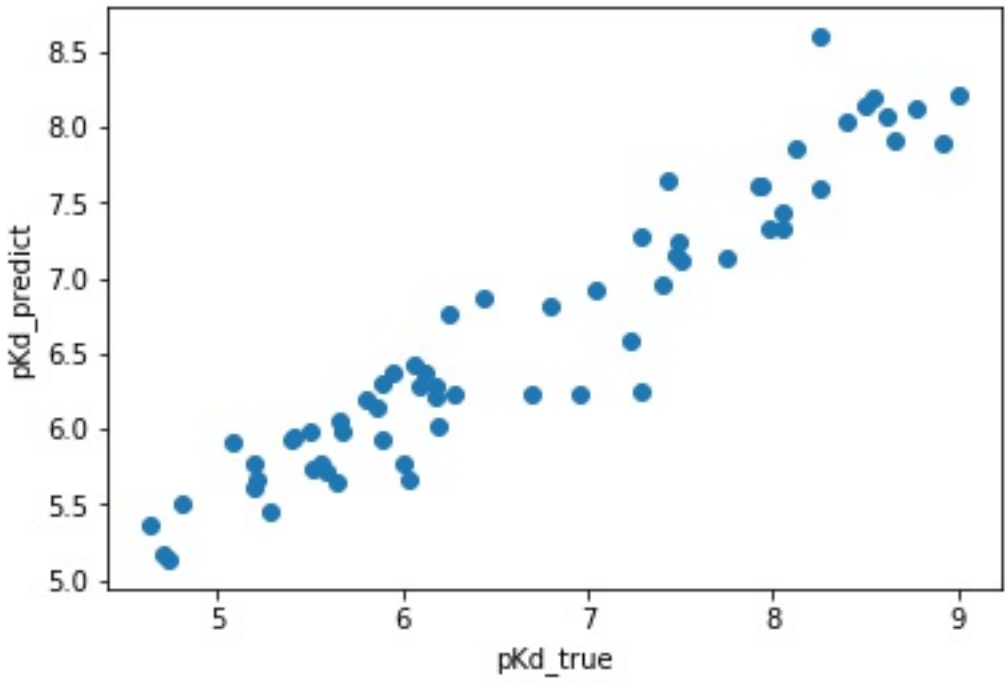


Fig. 1 plot of predicted pKd and reference pKd

pearson correlation coefficient	spearman correlation coefficient	Confidence interval	RMSE	F1	AUC
0.349646	0.293062	0.602083	1.169415	0.488121	0.677968

Table 1 Result of round 1b

RMSE	spearman correlation coefficient	AUC
1.287	0.224	0.594

Table 2 Result of round 2

Usage of code

A. Following is demo of constructing docker of N121 and run docker of N121

(A) Download "data" folder (let the location be C:\idg\data)

(B) In cmd

```
cd C:\idg\data
docker build -t  docker.synapse.org/syn18507261/n121_idg:9686281 .
docker login -u <user_ID> -p <password> docker.synapse.org
docker push docker.synapse.org/syn18507261/n121_idg:9686281
docker run -it --rm -v C:/idg/data/io:/output
```

```
docker.synapse.org/syn18507261/n121_idg:9686281
```

(C) local directory io has template.csv and after running will contain predictions.csv

References

1. Tang, J., et al., Drug Target Commons: a community effort to build a consensus knowledge base for drug-target interactions. Cell chemical biology, 2018. 25(2): p. 224-229. e2.
2. G, L. RDKit: Open-source cheminformatics. . Available from: http://www.rdkit.org.
3. Rogers, D. and M. Hahn, Extended-connectivity fingerprints. Journal of chemical information and modeling, 2010. 50(5): p. 742-754.
4. Drewry, D.H., et al., Progress towards a public chemogenomic set for protein kinases and a call for contributions. PloS one, 2017. 12(8): p. e0181585.

Authors Contribution Statement

Chih-Han Huang, Edward S. C. Shih, Tsai-Min Chen Ming-Jing Hwang developed the model, Chih-Hsun Wu, Wei-Quan Fang, Jhih-Yu Chen assisted the research.

Team MLmed: A CNN for drug kinase binding prediction

Matthias Lienhard, Paul Prasse, Ivo Bachmann, Julia Ganzlin, Gal Barel, and Ralf Herwig

Abstract

For the drug kinase binding prediction DREAM challenge, we trained a deep convolutional neuronal network with the DTC dataset and three features for the proteins as well as three features for the chemical compounds as input. Hyperparameters (e.g. number and size of hidden layers) were determined by random search on a parameter grid.

Training data and features

As suggested by the challenge organisers, training data was extracted from the DTC dataset (Jing Tang, Balaguru Ravikumar, Zaid Alam, Anni Rebane, Markus Vähä-Koskela, Gopal Peddinti, Arjan J van Adrichem, Janica Wakkinen, Alok Jaiswal, Ella Karjalainen, et al. Drug target commons: a community effort to build a consensus knowledge base for drug-target interactions. Cell chemical biology, 25(2):224–229, 2018). To describe the proteins, we used the protein sequence, the species, and presence of selected protein domains as features. Features describing compounds are the canonical SMILES sequence, the MACCS 166 keys (MDL Information Systems. Maccs keys.) and a 1024 bit molecular fingerprint computed by the RDKit chemoinformatics software (Open-source cheminformatics. http://www.rdkit.org).

Filtering and preprocessing of the DTC dataset

We selected Inchi keys and Uniprot ids as primary identifiers for the compounds and target proteins, respectively. For the DTC data, we first complemented missing Inchi keys by requesting Chembl using the compound ids. Next we filtered the data by value type for Kd and similar values (`standard_type` in ["KD","Kd","KI","Ki","pKD"]), concentration units (`standard_units` in ["NM", "MM", "UM", "M", "NMOL/L"]). We further excluded measurements on protein complexes, and requested the relation to be equal (`standard_relation` '='). Then we transformed all concentrations to pKd[M]. Mutated proteins were kept, but to avoid ambiguity of the protein id, we added the mutation descriptions to the uniprot id. Finally, we exported the relevant columns, the protein and compound id as well as the pKD[M] transformed value as tab separated text for easy import. From these dataset, we used two different subsets to train our models: a first subset *complete* containing all data points from the dataset described above and a second subset *human kinases* containing only the data points involving human kinases. Details of the used datasets can be seen in Table 1

dataset	# instances	# unique compounds	# unique proteins
complete	2,185,412	821,046	6,032
human kinases	215,697	110,431	467

Table 1: Training datasets

Kinase features

For the Proteins we considered three different features: the protein sequence, the species, and the presence of different kinase domains. We obtained the protein sequence as well as the species from a uniprot fasta file as well as

web requests for proteins missing in the fasta. For mutated proteins, we altered the protein sequence according to the mutation. Then we ran InterProScan (Philip Jones, David Binns, Hsin-Yu Chang, Matthew Fraser, Weizhong Li, Craig McAnulla, Hamish McWilliam, John Maslen, Alex Mitchell, Gift Nuka, et al. Interproscan 5: genome-scale protein function classification. Bioinformatics, 30(9):1236–1240, 2014.) on the protein sequence to determine the domain structure. We selected domains related to GO:0004672 ('protein kinase activity') which were present in at least 100 proteins from the DTC dataset to be considered.

Compound features

The three features we considered for the chemical compounds were: the SMILES sequence, the MACCS 166 keys and a 1024 bit molecular fingerprint. In order to obtain consistent features, we used the RDKit chemoinformatics software to produce and canonicalize SMILES. The same software was used to produce the fingerprint and the MACCS keys.

Model description

Model architecture

We used a deep neuronal network architecture to predict the target value. For the protein sequences as well as the SMILES we used one to three convolutional layers, followed by one to three dense hidden layers. The remaining features, e.g. the protein domains and the species corresponding to the proteins, as well as the MACCS keys and 1024 bit molecular fingerprint for the compounds, were modelled by one to three dense layers. The output of all 6 feature networks were concatenated and integrated by one to three hidden layers, resulting in a single output node representing the target value. The overall network is shown in Figure 1. Our model framework is implemented in Python, based on Keras (François Chollet et al. Keras. https://keras.io, 2015.) and Tensorflow (Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.) libraries as well as the scikit-learn (F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. Journal of Machine Learning Research, 12:2825–2830, 2011.) machine learning package. We trained the networks on a single server with a 40-core Intel(R) Xeon(R) CPU E5-2640 processor, 128 GB of memory and GeForce GTX TITAN X GPU using the NVidia CUDA platform.

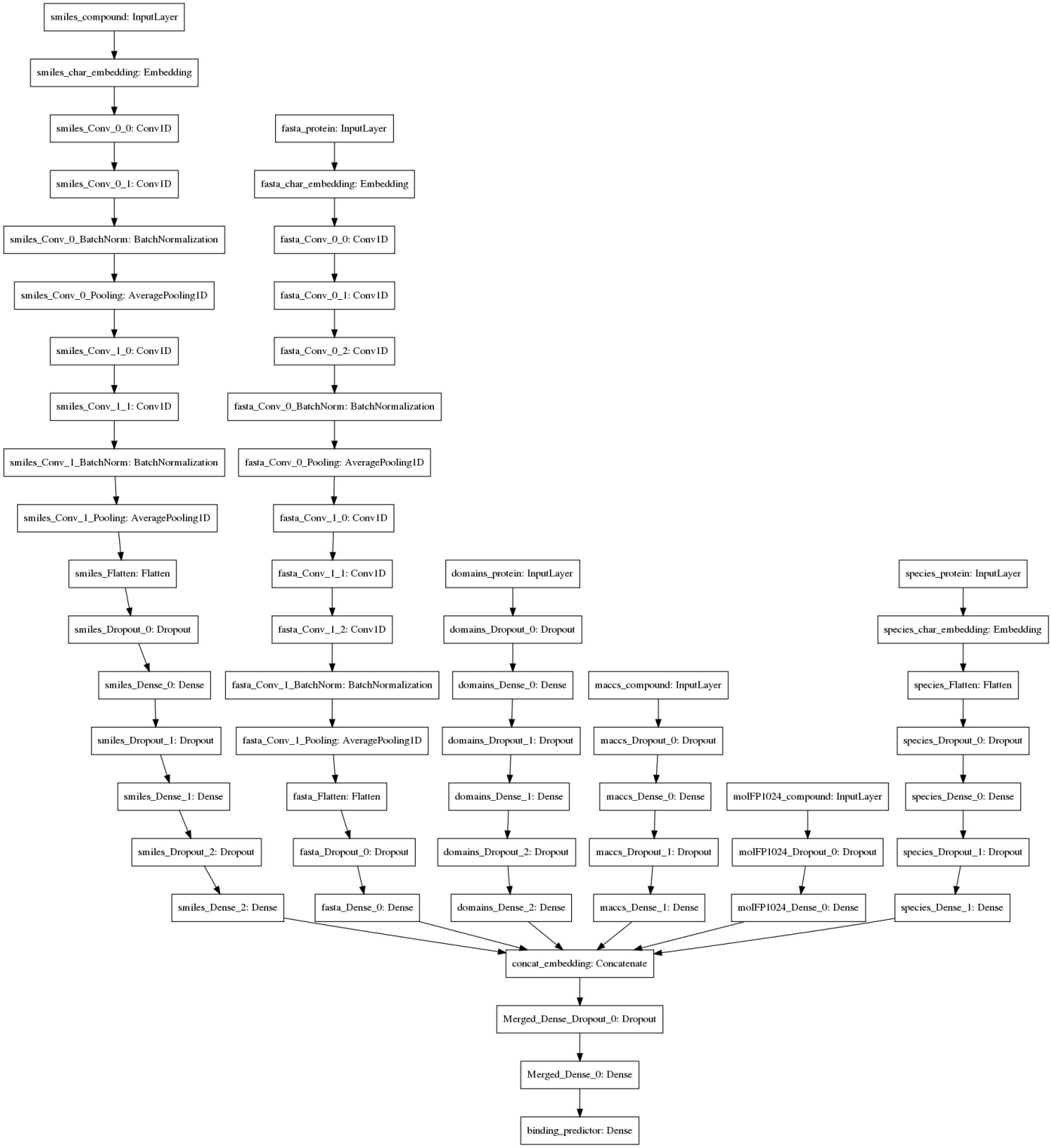


Figure 1: Final network architecture

Hyperparameter fitting

To find the best working parameters for our CNN we performed a random search on the paramter grid shown in Table 2. To score our models we performed a test on a hold-out-dataset containing a small proportion of compounds not present in the training data. For the round 2 submission we selected the best performing model trained on the *human kinases* dataset as well as the best model trained on the *complete* dataset (see Table 1)

hyperparameter	range
fasta embedding size	$\{2^x \mid x \in \{3 \dots 7\}\}$
fasta hidden size	$\{2^x \mid x \in \{3 \dots 7\}\}$
fasta filter size	$\{2^x \mid x \in \{5 \dots 7\}\}$
fasta kernel size	{3, 6, 12, 24, 48}
fasta num cnn layer	{1, 2, 3}
fasta num hidden layer	1, 2, 3
smiles embedding size	$\{2^x \mid x \in \{3 \dots 7\}\}$
smiles hidden size	$\{2^x \mid x \in \{3 \dots 7\}\}$
smiles filter size	$\{2^x \mid x \in \{5 \dots 7\}\}$
smiles kernel size	{3, 6, 12, 24, 48}
smiles num cnn layer	{1, 2, 3}
smiles num hidden layer	{1, 2, 3}
species embedding size	$\{2^x \mid x \in \{3 \dots 7\}\}$
species hidden size	$\{2^x \mid x \in \{3 \dots 7\}\}$
species num hidden layer	1, 2, 3
maccs hidden size	$\{2^x \mid x \in \{3 \dots 7\}\}$
maccs num hidden layer	1, 2, 3
molFP1024 hidden size	$\{2^x \mid x \in \{3 \dots 7\}\}$
molFP1024 num hidden layer	1, 2, 3
domain hidden size	$\{2^x \mid x \in \{3 \dots 7\}\}$
domain num hidden layer	1, 2, 3
merged hidden size	$\{2^x \mid x \in \{3 \dots 10\}\}$
merged num hidden layer	1, 2, 3

Table 2: Range for hyperparameters

Prediction using Docker

To create the predictions of submission with objectId 9686208 run:

```
docker run -v ${PWD}/io/:/input/ -v ${PWD}/io/:/output/
docker.synapse.org/syn18502700/ml-med:9686208
```

To create the predictions of submission with objectId 9686266 run:

```
docker run -v ${PWD}/io/:/input/ -v ${PWD}/io/:/output/
docker.synapse.org/syn18502700/ml-med:9686266
```

Experimental Results

On held out validation data from DTC the models are able to predict the target value and reaching a RMSE under 0.8, and $\varnothing$ AUC over 0.9 (Table 3). These results stand in marked contrast to the performance of our submissions.

train data	validation data	RMSE	pearson	spearman	f1	avg AUC
complete	*complete*	0.76	0.84	0.8	0.74	0.94
complete	*human kinases*	0.85	0.77	0.76	0.71	0.89
complete	*round 2*	1.19		0.353		0.696
human kinases	*complete*	1.34	0.35	0.3	0.29	0.78
human kinases	*human kinases*	0.74	0.84	0.83	0.78	0.92
human kinases	*round 2*	1.154		0.213		0.618

Table 3: Validation results for best models trained on *complete*and *human kinases*datasets.

Conclusions

Provided the performance on the held out validation data, we are disappointed by the outcome on the challenge dataset. Due to the restrictive challenge rules only allowing for a very limited number of submissions and obscuring the metrics, we were unable to find the cause of this difference in performance. We hope for insight from the solutions of the other teams and from feedback during the community phase of the challenge.

Authors' Contributions

M.L. and P.P. developed, implemented, and trained the model. I.B., J.G., and G.B. created features. R.H. supervised the project.

LET_DATA_TALK team IDG-DREAM challenge submission

Xiaokang Wang[1], Marouen Ben Guebila[2], Behrouz Shamsaei[3], Sourav Singh[4].
[1]Department of Biomedical Engineering, University of California, Davis.
[2]Department of Biostatistics, Harvard School of Public Health.
[3] Department of Environmental Health, College of Medicine, University of Cincinnati
[4]Department of Computer Engineering, VIIT, Pune

Abstract

The prediction of compound affinity towards protein targets is of paramount importance in the drug discovery and development process. The IDG-DREAM challenge addressed the question of the prediction of compound-protein kinase affinity as measured by pKD using machine learning approaches and public databases. The approach of LET_DATA_TALK consisted of building a regressor for each protein kinase and learning the affinity regression parameters for all its known interaction partners using compound features. Our approach had a rounded RMSE of 1.372, a rounded spearman of 0.33, and a rounded average AUC of 0.699.

Introduction

Preclinical drug development requires the design and optimization of novel candidate molecules endowed with bioactive properties to treat or decrease the progress of human diseases. The selection of compound in early stages of development is based on the screening of large libraries. The development of computational methods aided the automatic screening of compounds against targets of interest, particularly the tyrosine kinase family of proteins that are involved in several cancers (Arora and Scholar 2005).

The IDG-DREAM challenge consisted of predicting the pKD affinity value for pairs of protein kinase and compounds. The general approach was to collect features about protein kinases and their inhibitors using publicly accessible databases such as the DTC (Tanoli et al. 2018), PubChem (Kim et al. 2016), and Chembl (Gaulton et al. 2012). Consequently, a machine learning model is trained on the computed features to predict the affinity of the target-compound pair in the test set.

The baseline example provided in the challenge was based on a publication from the challenge organizers (Cichonska et al. 2017). The approach consisted of crafting a large set of features for each protein and compound in the training set including the protein sequence, the protein tridimensional conformation, the kinase binding site sequence, protein-protein similarity scores, compound chemical structure, and compound molecular weight. The features are used to train a pairwise regression kernel for each drug-compound pair. The baseline method achieved in round 1a a rounded RMSE of 1.2821, a rounded spearman of 0.4052, and a rounded average AUC of 0.3757. In round 2, it achieved a rounded RMSE of 1.123, a rounded spearman of 0.401, and a rounded average AUC of 0.72.

LET_DATA_TALK presented a method that builds a regressor for each protein, thus requiring only the features of the compounds. The drug molecular structure is converted into a fingerprint containing features that are associated with the pKD values. The drug-protein parameters were extracted from the DTC (Tanoli et al. 2018) to train the machine learning models. Encouragingly, in round 1a, our submission did better than the baseline with a rounded RMSE of 1.4056, a rounded spearman of 0.3203, and a rounded average AUC of 0.3576. In round 2, our submission had a rounded RMSE of 1.372, a rounded spearman of 0.33, and a rounded average AUC of 0.699.

Data

The data we used were downloaded from DTC [ ] and only records/rows that have a measured Kd, KD, KDAPP, Ki, KI, or KI RATIO. Ki was treated as equivalent to Kd as we observed a boost in performance from 0.33 to 0.42 in terms of Pearson correlation coefficient in round 1. Thus we stuck to this in round 2. We also downloaded all the measured pairs of protein and chemicals in the chembl database, which shew a high overlap with the data in DTC. Only records related to the proteins in the testing set were included as we built a model for each protein (see methods part). No training data was available for 4 kinases in the testing set. In total, we had 101,469 unique protein-chemical pairs. The distribution of the number of records for each protein is shown in Fig. 1. 143 kinases out of the 203 kinases have 200 or more records. The median was treated as the truth if there are replicates given a pair. We tried to use other measured activities, e.g. IC50, as a feature when predicting kd but most of the pairs of protein and chemicals that have a measured Kd don't have measured other activities.

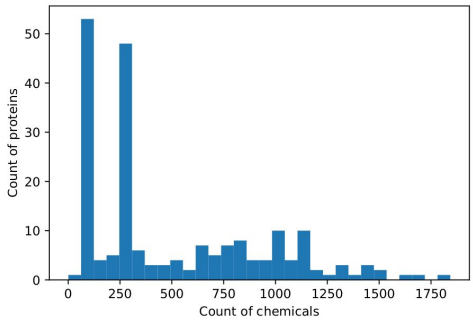


Fig. 1. The distribution of the number of chemicals in the training set for each protein in the testing set.

Methods

We tried to build a single model to predict the activation given a pair of protein and chemical. A protein was treated as a sequence of amino acids. Each adjacent three amino-acids were encoded by a numeric embedding, which was published in []. A chemical was represented by a fingerprint (FP). The types of FP we tested include Morgan FP and Topological Torsion FP in the rdkit package. Each FP is a vector of 1024 binary variables. To capture the interaction

between a kinase and a chemical, we built a mixture of feedforward neural network (FNN) and recurrent attention network (RAN). Specifically, the FP of a chemical was fed into an FNN and the protein sequence was fed to RAN, and then the outputs from both networks were merged by two layers of fully connected layers followed by a one-node regression layer. We conducted 10-fold cross-validation on the training dataset and the performance is 0.58 in terms of Spearman correlation coefficient.

Another approach we explored is to build a model for each protein considering that we don't have to model the complex structure of a protein. One drawback of this approach is that the records that are not related to the kinases in the testing set were excluded. However, we got even better performance in cross-validation than the first approach. We did not compare these two approaches on the testing set in round 2. For this approach, the model we built is support vector regression and kernel regression model.

Results and Conclusion

We observed a strong gap between the performance on the validation set and that on the testing set. One possible reason is that the model was overfitted on the validation set when tweaking the hyper-parameters in a SVR model. The two hyper-parameters in SVR are c and gamma, which controls the width of the soft-margin, which is also reversely related to the cost of misclassifying a data point, and the locality of a support vector, respectively. A larger c and large gamma might raise the alarm of overfitting. But we did not test this in round 1. Another possible reason is that there are very similar pairs of chemical and protein in the training set. Thus the leave-out set is similar to the training set in CV, whereas the testing data set differs from the training set. At this point, we are open to these reasons and other possible reasons.

Table 1. The performance of the kinds of models in 5-fold cross validation trained on the Morgan FP and Topological FP. A tuple in each cell denotes Pearson correlation coefficient, Spearman correlation coefficient and mean absolute error, respectively.

	Morgan	Topological
Support vector regression	(0.79, 0.70, 0.43)	(0.78, 0.70, 0.43)
Kernel regression	(0.79, 0.69, 0.42)	(0.79, 0.68, 0.43)

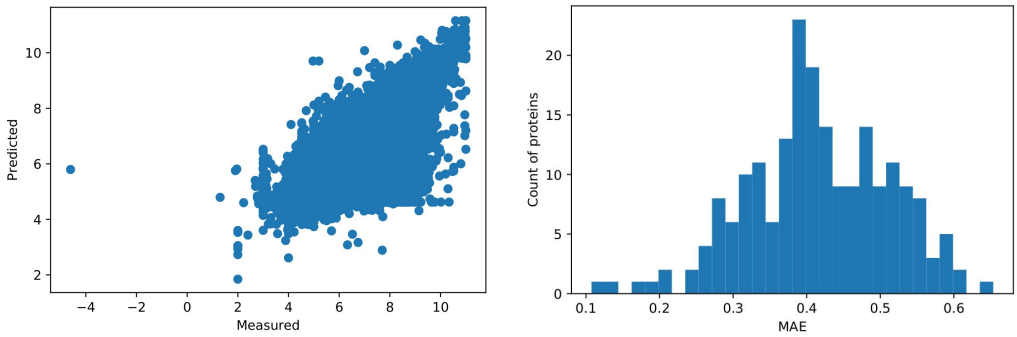


Fig. 2. Visualization of the measured value and prediction in CV on the training set. The distribution of the MAE of the model for each protein in CV.

Author contributions

All the authors participated in round 1. Xiaokang and Marouen lead in the round 2.

References

Arora, Amit, and Eric M. Scholar. 2005. “Role of Tyrosine Kinase Inhibitors in Cancer Therapy.” *The Journal of Pharmacology and Experimental Therapeutics* 315 (3): 971–79.

Cichonska, Anna, Balaguru Ravikumar, Elina Parri, Sanna Timonen, Tapio Pahikkala, Antti Airola, Krister Wennerberg, Juho Rousu, and Tero Aittokallio. 2017. “Computational-Experimental Approach to Drug-Target Interaction Mapping: A Case Study on Kinase Inhibitors.” *PLoS Computational Biology* 13 (8): e1005678.

Gaulton, Anna, Louisa J. Bellis, A. Patricia Bento, Jon Chambers, Mark Davies, Anne Hersey, Yvonne Light, et al. 2012. “ChEMBL: A Large-Scale Bioactivity Database for Drug Discovery.” *Nucleic Acids Research* 40 (Database issue): D1100–1107.

Kim, Sunghwan, Paul A. Thiessen, Evan E. Bolton, Jie Chen, Gang Fu, Asta Gindulyte, Lianyi Han, et al. 2016. “PubChem Substance and Compound Databases.” *Nucleic Acids Research* 44 (D1): D1202–13.

Tanoli, Ziaurrehman, Zaid Alam, Markus Vähä-Koskela, Balaguru Ravikumar, Alina Malyutina, Alok Jaiswal, Jing Tang, Krister Wennerberg, and Tero Aittokallio. 2018. “Drug Target Commons 2.0: A Community Platform for Systematic Analysis of Drug-Target Interaction Profiles.” *Database: The Journal of Biological Databases and Curation* 2018 (January): 1–13.

Two-step kernel ridge regression to predict drug-kinase interactions for the IDG-DREAM Drug-Kinase Binding Prediction Challenge

Introduction

Recently, the Research Unit Knowledge-Based Systems (KERMIT) has developed software for a two-step kernel ridge regression method that can be used in a variety of pairwise learning settings. The methods are implemented in the 'xnet' R package and are available via GitHub (https://centerforstatistics-ugent.github.io/xnet/). The IDG-DREAM Drug-Kinase Binding Prediction Challenge represented an ideal opportunity to test this newly developed software package.

Materials and methods

In a first step, the provided raw dataset was processed. Only data that were annotated with 'K DISS', 'LOGKD', '-LOG K', 'KDISS', 'KD', 'LOG K', '-LOG KDISS', '-LOG KD', 'LOG KD', 'Kd' or 'KD' were kept in the dataset. Furthermore, data points having an unspecified compound or target were deleted from the dataset. Secondly, for drug-kinase interactions having multiple measurements, the average of these measurements was computed and other measurements were deleted. In total, this yielded 42730 measured drug-kinase interactions from which machine learning models could learn patterns. More specifically, a two-step kernel ridge regression method was used (Stock *et al.*, 2018). Therefore, kernel matrices were computed that represented the drugs and kinases. For the drugs, molecular fingerprints were calculated based on the provided SMILE representations of the molecules. This was done using RDKit in Python. From these molecular fingerprints, the Tanimoto similarity measure was calculated to construct a kernel matrix. Additionally, a Gaussian interaction profile (GIP) kernel matrix was computed using the Tanimoto similarities (for imputation of missing interaction values) and the scikit-learn toolbox in Python. Both kernel matrices were transformed to a positive semi-definite matrix by iteratively adding small constants to the first diagonal of the matrix. The Tanimoto kernel matrix and GIP kernel matrix were combined to one final drug kernel matrix. Protein kinases were represented using a kernel matrix constructed from pairwise alignment scores. The computed kernel matrices were used as input for a two-step kernel ridge regression model, as was implemented in the 'xnet' R package (https://centerforstatistics-ugent.github.io/xnet/). Using leave-one-out cross validation, the two hyperparameters (one for each kernel matrix) were optimized. Afterwards, a final model was built using these optimized values and was used to make predictions with.

Results and conclusion

Although it was expected for the method and computed kernel matrices to perform well in this setting, results were unsatisfactory. Performances for predictions in round 2 were as follows: an RMSE of 8.466, a rounded spearman correlation of 0.147 and a rounded average AUC of 0.54. In retrospect, computing averages for the interactions occurring multiple times might not have been a good choice. In addition, the transformation to positive semi-definite matrices could have negatively affected the feature representation. In conclusion, although the implemented R package is easy to work with and is useful in a variety of pairwise prediction settings, the method used for this challenge did not yield adequate results. Improvement of the data preprocessing steps and kernel computation steps can improve performance in the future.

Authors contribution statement

Michiel Stock designed the two-step kernel ridge regression methods. Dimitri Boeckaerts processed data for this challenge, implemented scripts for computation of feature representations, used the implemented methods (xnet R package) to train and test a two-step kernel ridge regression model and analysed results as to find the best performing model. Bernard De Baets and Yves Briers provided funding for this project.

References

Stock, M., Pahikkala, T., Airola, A., Waegeman, W. and De Baets, B. Algebraic shortcuts for leave-one-out cross-validation in supervised network inference. Briefings in Bioinformatics. Stock, M., Pahikkala, T., Airola, A., De Baets, B. and Waegeman, W. A comparative study of pairwise learning methods based on kernel ridge regression. Neural Computation 30 (2018), 2245-2283.

Instructions to run the docker (source code under Files in src directory)

```
docker run -it --rm -v ${PWD}/io:/input -v ${PWD}/io:/output
docker.synapse.org/syn18500740/tskrr-dock:9686206
```

DMIS_DK Submission

Sungjoon Park[1], Minji Jeon[1], Sunkyu Kim[1], Junhyun Lee[1], Seongjun Yun[1], Bumsoo Kim[1], Buru Chang[1], and Jaewoo Kang[1,2,*] 1.Department of Computer Science and Engineering, Korea University, Seoul, Republic of Korea 2.Interdisciplinary Graduate Program in Bioinformatics, Korea University, Seoul, Republic of Korea

[*] corresponding author

Introduction

It is important to obtain binding affinity between drugs and kinases in drug discovery process. However, measuring binding affinity is cost-intensive. To address this, we developed a machine learning model to predict binding affinity between drugs and kinases. The IDG-DREAM Drug-Kinase Binding Prediction Challenge provided the UniProt IDs of proteins in the Drug Target Commons dataset and we could get sequence information of the proteins using the UniProt IDs. In the binding affinity prediction task, 3D structural information of proteins is known to be more informative than sequence information of proteins. However, predicting of structures of proteins is a difficult problem. We, therefore, propose a ligand-based prediction model that focuses on the structures of drugs, rather than using information such as sequences of proteins or structures of proteins.

Methods

Data

In this challenge, we used Drug Target Commons (DtcDrugTargetInteractions.csv) [1] and BindingDB data (BindingDB_All_2019m2.tsv.zip) [2] for training binding affinity prediction models. Among diverse measurements, kd, ki and IC50 were used for the training. All binding affinity values were transformed by -log10(x/1e9). We chose median values for the duplicate samples (i.e., same compound-protein pair). We submitted two prediction models: Random Forest (RF) and Ensemble of multi-task Graph Convolutional Networks (GCN). For the RF model, 128,181 samples (compound-protein pair) having 199 proteins and 57,399 compounds were used for the training. For the multi-task GCN model, 953,521 samples (compound-protein pair) having 1,474 proteins and 474,875 compounds were used for the training.

Models

Random Forest

We trained a Random Forest model [3] for each protein. 2048-dimensioned Extended Connectivity Fingerprint (ECFP) [4] is used as input. ECFP is one of the drug structure representation methods that represents the presence of substructures in a molecule as a binary vector. The output of the model is the pKd, pKi and IC50 values of the dataset. We divided the dataset into 80:20 and used each dataset as a training set and a validation set. Hyper-parameters were selected based on the performance of the validation set. There is no model for S4 samples because we trained the models only for the proteins in the training set. For each S4 sample in Round 2, we measured the similarity between 199 proteins and the protein of the sample based on protein sequences, and selected the top 3 similar proteins. The predicted pKd value of the sample is the average of the predicted values from the top 3 protein models.

Multi-task GCN Ensemble

We designed 4 multi-task GCN architectures. The multi-task GCN model takes a SMILES string as input and predict binding affinities for 1,474 proteins. In the 1,474 proteins, 199 out of 207 round 2 proteins were included. SMILES strings were converted to molecular graphs using RDKit python library [5]. We designed a 78 dimensional feature

vector to represent a node (here, atom) in a molecular graph. Description of the feature vector is shown in Table 1. For the submission, we averaged the predictions of the last K epochs. Then, we averaged all the 12 multi-task GCN models (4 different architecture with 3 different weight initialization) averaged predictions. We selected the hyper-parameters of the multi-task GCN models based on the performance of the validation set. We implemented the GCN models using PyTorch Geometric (PyG) library. Procedure for predicting S4 samples in Round2 data was the same as the random forest model.

Multi-task GCN architecture 1

GAT layer + GCN layer + Pooling layer + 1 dense layer + 1 output layer GAT layer: Graph convolution layer using graph attention networks proposed in “Graph Attention Networks” [6]. Multi-head vectors were concatenated (# of head: 10, input dim: 78 , output dim: 780). GCN layer: Graph convolution layer proposed in "Semi-Supervised Classification with Graph Convolutional Networks" [7] (input dim: 780 , output dim: 780). Pooling layer: Concatenation of average pooling and max pooling across all node feature vectors (input dim: 780 , output dim: 1,560). Dense layer: Fully connected layer with dropout (dropout rate : 0.5, input dim: 1,560, output dim: 1,500). Output layer: Fully connected layer (input dim: 1,500 , output dim: 1,474).

Multi-task GCN architecture 2 /3

4 GCN layer & Pooling layer (after each GCN layer) + 4 GCN layer(after pooling) + 1 dense layer + 1 output layer GCN layer: Graph convolution layer proposed in "Weisfeiler and Leman Go Neural: Higher-order Graph Neural Networks" [8] (input dim: 78 , output dim: 128). Pooling layer: Hierarchical graph pooling layer proposed in “Self-Attention Graph Pooling” [9] (pooling ratio=0.25) GCN layer (after pooling): Graph convolution layer proposed in "Weisfeiler and Leman Go Neural: Higher-order Graph Neural Networks". (input dim: 128 , output dim: 128). Dense layer: Fully connected layer with dropout (dropout rate : 0.5, input dim: 512, output dim: 512). Output layer: Fully connected layer (input dim: 512 , output dim: 1,474).

Multi-task GCN architecture 4

4 GAT layer & Pooling layer (after each GCN layer) + 4 GCN layer(after pooling) + 1 dense layer + 1 output layer GAT layer: Graph convolution layer using graph attention networks proposed in “Graph Attention Networks”. Multi-head vectors were concatenated (# of head: 2 & 4 , input dim: 78 , output dim: 128). Pooling layer: Hierarchical graph pooling layer proposed in “Self-Attention Graph Pooling” (pooling ratio=0.25) GCN layer (after pooling): Graph convolution layer proposed in "Weisfeiler and Leman Go Neural: Higher-order Graph Neural Networks". (input dim: 128 , output dim: 128). Dense layer: Fully connected layer with dropout (dropout rate : 0.5, input dim: 512, output dim: 512). Output layer: Fully connected layer (input dim: 512 , output dim: 1,474).

Atom feature type	RDkit function	Encoding type	Dimension
Atom symbol	atom.GetSymbol()	One hot encoding	44
Degree	atom.GetDegree()	One hot encoding	11
Total number of Hs	atom.GetTotalNumHs()	One hot encoding	11
Implicit valence	atom.GetImplicitValence()	One hot encoding	11
Is aromatic	atom.GetIsAromatic()	Bool (0 or 1)	1
Total			78

Table 1. Description of the atom feature

Results

The results of round2 leaderboard

Model	objectID	RMSE	Spearman	AUC
Random Forest	9686312	1.002	0.484	0.774
Multi-task GCN Ensemble	9686330	0.949	0.485	0.771

References

[1] Tang, J., Ravikumar, B., Alam, Z., Rebane, A., Vähä-Koskela, M., Peddinti, G., ... & Gautam, P. (2018). Drug Target Commons: a community effort to build a consensus knowledge base for drug-target interactions. Cell chemical biology, 25(2), 224-229.

[2] Gilson, M. K., Liu, T., Baitaluk, M., Nicola, G., Hwang, L., & Chong, J. (2015). BindingDB in 2015: a public database for medicinal chemistry, computational chemistry and systems pharmacology. Nucleic acids research, 44(D1), D1045-D1053.

[3] Breiman, L. (2001). Random forests. Machine learning, 45(1), 5-32.

[4] Rogers, D., & Hahn, M. (2010). Extended-connectivity fingerprints. Journal of chemical information and modeling, 50(5), 742-754.

[5] RDKit: Open-source cheminformatics; http://www.rdkit.org

[6] Veličković, P., Cucurull, G., Casanova, A., Romero, A., Lio, P., & Bengio, Y. (2017). Graph attention networks. arXiv preprint arXiv:1710.10903.

[7] Kipf, T. N., & Welling, M. (2016). Semi-supervised classification with graph convolutional networks. arXiv preprint arXiv:1609.02907.

[8] Morris, C., Ritzert, M., Fey, M., Hamilton, W. L., Lenssen, J. E., Rattan, G., & Grohe, M. (2018). Weisfeiler and Leman Go Neural: Higher-order Graph Neural Networks. arXiv preprint arXiv:1810.02244.

[9] Lee, J., Lee, I., & Kang, J. (2019). Self-Attention Graph Pooling. arXiv preprint arXiv:1904.08082.

DeepAffinity: interpretable deep learning of compound–protein affinity through unified recurrent and convolutional neural networks

Mostafa Karimi, Di Wu, Zhangyang Wang, and Yang Shen

Department of Electrical and Computer Engineering, TEES–AgriLife Center for Bioinformatics and Genomic Systems Engineering and Department of Computer Science and Engineering, Texas A&M University, College Station, TX 77843, USA

Abstract

A semi-supervised deep learning model that unifies recurrent and convolutional neural networks [1] has been developed to exploit both unlabeled and labeled data, for jointly encoding molecular representations and predicting affinities. They are trained over generic protein-ligand data from BindingDB [19] and not fine-tuned for the kinase targets in the challenge.

Introduction

It is critically important to characterize compound–protein interaction (CPI) for drug discovery and development [2]. Considering the enormous chemical and proteomic spaces, computational prediction of CPIs facilitates experimental parallels and accelerates drug discovery. Indeed, computational prediction of CPI has made much progress recently, especially for repurposing and repositioning known drugs for previously unknown but desired new targets [3,4] and for anticipating compound side-effects or even toxicity due to interactions with off-targets or other drugs [5,6].

Computational methods roughly fall in two categories based on input data types: (protein) structure-based and sequence based methods. Structure-based methods can predict compound–protein affinity, i.e. how active or tight-binding a compound is to a protein; and their results are highly interpretable. They are often tackled through energy models [7] or machine learning [8,9]. Their heavy reliance on actual 3D structures of CPI presents a limitation for these methods. Sequence-based methods overcome the limited availability of structural data and the costly need of molecular docking. Rather, they exploit rich omics-scale data of protein sequences, compound sequences. Sequence-based CPI has been tackled through shallow models [10] or deep learning models [11,12] but their predictions lack interpretability.

To overcome limitations of current structure- and sequence-based CPI prediction methods, we have designed informative yet compact data representations that are structurally interpretable. We have also developed semi-supervised deep learning models that unify recurrent and convolutional neural networks, exploit labeled and unlabeled data, and use attention mechanisms for interpretability.

Methods

Data

We used data from three public datasets: all K_d labeled compound-protein binding data (17,819 samples) from BindingDB [19], compound data (500K samples for training and 500K samples for validation) in the SMILES format from STITCH [20] and protein amino-acid sequences from UniRef with 50% sequence identity and length less than or

equal to 1500 amino acids (120,000 samples for training and 50,525 for validation) [21] for training our unified RNN-CNN model.

Input formats

We developed a novel protein representation, Structural property sequence (SPS) by incorporating the predicted protein structural property such as secondary structure elements (SSEs), Solvent accessibility, physicochemical characteristics and length of each secondary structure elements (SSEs). For drug representation, we used SMILE [13] that are short ASCII strings to represent compound chemical structures based on bonds and rings between atoms.

Deep learning methods

First, we encoded compound SMILES or protein SPS into representations, by unsupervised deep learning from unlabeled data from STITCH and UniRef. Specifically, we used a recurrent neural network (RNN) model, seq2seq [14] that has seen much success in natural language processing and was recently applied to embedding compound SMILES strings into fingerprints [15]. We choose gated recurrent unit (GRU) [16] with attention mechanism [17] as our seq2seq model.

Next, with compound and protein representations learned from the above unsupervised learning, we solve the regression problem of compound–protein affinity prediction using supervised learning. For either proteins or compounds, we append a CNN after the RNN (encoders and attention models only) that we just trained. The CNN model consists of a one-dimensional (1D) convolution layer followed by a max-pooling layer. The outputs of the two CNNs (one for proteins and the other for compounds) are concatenated and fed into two more fully connected layers. The entire RNN-CNN pipeline is trained from end to end [18], with the pre-trained RNNs serving as warm initializations, for improved performance over two-step training. More details about how the final models are derived are included in the next subsection.

Lastly, we have also introduced protein and compound attention models in supervised learning to both improve predictive performances and enable model interpretability at the level of letters (SSEs in proteins and atoms in compounds). In the supervised model we just have the encoder and its attention α_t on each letter t for a given string x (protein or compound). And the output of the attention model, A, will be the input to the subsequent 1D-CNN model. Suppose that the length of protein encoder is T and (s_1,..,s_t,..., s_T) are the output of protein encoder and similarly the length of compound encoder is D and $$(m_{1},..,m_{_d},..., m_D) $$ are the output of compound encoder. We parametrize the attention model of unified model with matrix $$$U_a$$$ and the vector $$$v_a$$$. Then, The attention model is formulated as:

$$e_t = v_a tanh(W_a s_t) \quad \forall t = 1, \cdots, T \text{ where: } \alpha_t = \frac{exp(e_t)}{\sum_k exp(e_k)} \quad \forall t = 1 \cdots T \text{ and } A = \sum_t \alpha_t s_t.$$

The attention weights (scores) α_t suggest the importance of the t^{th} "letter" (secondary structure element in proteins and atom or connectivity in compounds) and thus predict the binding sites relevant to the predicted binding affinity.

Models submitted

We give more details about the training process for final models as follows. We trained three unified RNN-CNN models with different neurons (300,100), (400,200), and (600,300) at their fully connected layers. For each of these unified RNN-CNN model, we at first pre-trained the RNN encoder part from the encoder part of our seq2seq model and fixed the encoder parts. We trained the rest of the architecture with Adam optimizer [22] with an initial learning rate of 0.001 for 100 epochs. Later, we jointly trained all the architecture with Adam optimizer with an initial learning rate of 0.0001 for another 100 epochs. Finally, motivated from ensemble methods, we consider the last 10 epochs of each model as a predictor. Finally, we take an average of all 30 predictors to calculate the final prediction. Our docker image and src directory provides the 3 unified models with 10 checkpoints (epochs) each.

Conclusion

We have developed accurate and interpretable deep learning models for predicting compound–protein affinity using only compound identities and protein sequences. By taking advantage of massive unlabeled compound and protein data besides labeled data in semi-supervised learning, we have jointly trained unified RNN-CNN models from end to end for learning context- and task-specific protein/compound representations and predicting compound–protein affinity. Given the novel representations with better interpretability, we have included attention mechanism in the unified RNN-CNN models to quantify how much each part of proteins, compounds, or their pairs are focused while the models are making the specific prediction for each compound–protein pair. Noting that our models submitted were trained over generic data, improvements can be made by tailoring and tuning the models for kinase targets.

How to run our image

```
docker run \-\-privileged=true -it --rm -v \$\{PWD\}/io:/input -v $\{PWD\}/io:/output
docker.synapse.org/syn17051692/deepaffinity
```

References

[1] Mostafa Karimi, Di Wu, Zhangyang Wang, Yang Shen, DeepAffinity: interpretable deep learning of compound–protein affinity through unified recurrent and convolutional neural networks, *Bioinformatics*, , btz111, https://doi.org/10.1093/bioinformatics/btz111 [2] Santos R.*et~al*. (2017) A comprehensive map of molecular drug targets. *Nat. Rev. Drug Discov.*, 16, 19–34. [3] Keiser M.J .*et~al*. (2009) Predicting new molecular targets for known drugs. *Nature*, 462, 175. [4] Power A. *et~al*. (2014) Genomics-enabled drug repositioning and repurposing: insights from an IOM Roundtable activity. *JAMA*, 311, 2063–2064. [5] Chang R.L .*et~al*. (2010) Drug off-target effects predicted using structural analysis in the context of a metabolic network model. *PLoS Comput. Biol.*, 6, e1000938. [6] Mayr A. *et~al*. (2016) Deeptox: toxicity prediction using deep learning. *Front. Environ. Sci.*, 3, 80. [7] Gilson M.K., Zhou H.-X.(2007) Calculation of protein–ligand binding affinities. *Annu. Rev. Biophys. Biomol. Struct.*, 36, 21–42. [8] Wallach I.*et~al*. (2015) Atomnet: a deep convolutional neural network for bioactivity prediction in structure-based drug discovery. arXiv Preprint arXiv: 1510.02855. [9] Gomes J.*et~al*. (2017) Atomic convolutional networks for predicting protein–ligand binding affinity. arXiv Preprint arXiv: 1703.10603. [10] Shi Y.*et~al*. (2013) Protein–chemical interaction prediction via kernelized sparse learning svm. In: *Pacific Symposium on Biocomputing*, pp. 41–52. [11] Tian K.*et~al*. (2016) Boosting compound–protein interaction prediction by deep learning. *Methods*, 110, 64–72. [12] Wan F., Zeng J. (2016) Deep learning with feature embedding for compound–protein interaction prediction. bioRxiv, 086033. [13] Weininger D. (1988) Smiles, a chemical language and information system. 1. introduction to methodology and encoding rules. J. *Chem. Inf. Comput. Sci.*, 28, 31–36. [14] Sutskever I.*et~al*. (2014) Sequence to sequence learning with neural networks. In: *Advances in Neural Information Processing Systems*, pp. 3104–3112. [15] Xu Z.*et~al*. (2017). Seq2seq fingerprint: an unsupervised deep molecular embedding for drug discovery. In: Proceedings of the 8th ACM *International Conference on Bioinformatics, Computational Biology, and Health Informatics*. ACM, pp. 285–294. [16] Cho K.*et~al*. (2014) On the properties of neural machine translation: encoder–decoder approaches. In: Proceedings of SSST-8, Eighth Workshop on *Syntax, Semantics and Structure in Statistical Translation*, Doha, Qatar, pp. 103–111. [17] Bahdanau, Dzmitry, Kyunghyun Cho, and Yoshua Bengio. "Neural machine translation by jointly learning to align and translate." arXiv preprint arXiv:1409.0473 (2014). [18] Wang Z.*et~al*. (2016b) Studying very low resolution recognition using deep networks. In: Proceedings of the IEEE Conference on *Computer Vision and Pattern Recognition*, pp. 4792–4800. [19] Liu T.*et~al*. (2006) Bindingdb: a web-accessible database of experimentally determined protein–ligand binding affinities. *Nucleic Acids Res.*, 35, D198–D201. [20] Kuhn M.et al. (2007) Stitch: interaction networks of chemicals and proteins. *Nucleic Acids Res.*, 36, D684–D688. [21] Suzek B.E.*et~al*. (2015) Uniref clusters: a comprehensive and scalable alternative for improving sequence similarity searches. *Bioinformatics*, 31, 926–932. [22] Kingma, Diederik P., and Jimmy Ba. Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014).

Authors Statement

All conceived the learning schemes; MK implemented all the deep learning models and trained them; DW managed data cleaning, and made the docker; YS, MK and DW wrote the wiki.

IDG-DREAM Drug-Kinase Binding Prediction Challenge: Group KinaseHunter

Hansaim Lim[1] and Lei Xie[1,2]
[1]Ph.D. program in Biochemistry, Graduate Center, The City University of New York
[2]Department of Computer Science, Hunter College, The City University of New York

Data set preparation

1. Chemical-kinase binding affinity data collection

To build a large-scale chemical-kinase binding affinity data set, we integrated multiple public databases and published kinome assays. The databases we used are ChEMBL (ver. 24), BindingDB, and LINCS-HMS KinomeScan database. ChEMBL database contains chemical-protein binding affinities for mutant proteins. We collected those activities for mutants from ChEMBL. BindingDB contains chemical-protein affinity data sets from multiple sources, including PubChem, PDSPKi, U.S. patents, and the curated data set by BindingDB team. LINCS-HMS KinomeScan database contains chemical-kinase binding affinities for various chemicals and target kinases. Four published kinome assay data sets were included from 1) Christmann-Franck et al. JCIM, 2016, 2) Drewry et al. PLoS One, 2017, 3) Klaeger et al. Science, 2017, and 4) Sorgenfrei et al. ChemMedChem, 2017. To merge redundant activity measurements (e.g. multiple activity records for a chemical-kinase pair), we converted chemicals into InChIKey and kinases into UniProt ID. We converted all activities into log-scale since the target problem is to predict binding affinity in pKd. We excluded activity measurements in other metrics than Ki, pKi, Kd, and pKd. If multiple activity records found for a chemical-kinase pair, we averaged them. For feature calculation processes, we collected SMILES strings for each chemical and primary amino acid sequences for each kinase. In case the data source does not provide chemical SMILES or InChIKey, we used the PUGRest service from PubChem to convert the chemical identifiers. We used UniProt database to convert gene names into UniProt IDs. For mutant proteins, we prepared mutated protein sequences by replacing, inserting, or deleting sequence parts as appeared in protein identifiers.

2. Splitting data into train, dev, and test sets

We used Leave-Chemical-Set-Out (LCSO) strategy to simulate new drug discovery process and reduce overfitting. To evaluate model fitting and generalized performance, we split the chemical-kinase samples into train, dev, and test sets. We used train set to train models, dev set to evaluate how well the model is trained and optimize hyperparameters, and test set to evaluate the final performance of our model. Before the final prediction for the challenge, we trained the model again with the dev and test sets.

We split the chemicals into two groups: 42708 training chemical set and 581 dev-test chemical set. We kept the structural similarity for any chemical-chemical pairs across two sets lower than 0.8, ensuring that the dev or test data represent new chemical molecules that are not found in training data. The chemical-chemical similarity scores were measured by Tanimoto coefficient (Jaccard similarity) between the two ECFP4 (1024 bits)

representations of chemical molecules. The chemical-kinase affinity samples for dev-test chemicals are split into dev and test sets in approximately 8:2 ratio. The data statistics for train/dev/test sets are in Table 1.

Table 1	Train	Dev	Test
#samples	400170	33701	8618
#unique chemicals	42708	520	343
#unique proteins	482	460	447

Method

1. Model architecture

Our model uses graph-based molecular representations for chemicals and proteins. Our target problem is to predict binding affinity in pKd, given the input chemical and kinase. We denote each sample of chemical-kinase affinity data as $y_i = (c_i, k_i)$, where y_i is the known pKd value, c_i is the chemical, and k_i is the kinase in the i^{th} sample.

Our model processes chemical molecules using graph convolutional neural network (Neural Fingerprint) proposed by Duvenaud et al. (NIPS, 2015), which produces chemical molecular fingerprint of a user-defined length. Briefly, the Neural Fingerprint applies weight filters to atoms and bonds ordered by their degrees in each molecule. We set the length of chemical fingerprint to 128. We denote the chemical fingerprint operation as $f_{c_i} = \mathcal{G}_c(c_i)$, where f_{c_i} is the feature vector representation of c_i ($f_{c_i} \in \mathbb{R}^{128}$), and $\mathcal{G}_c$ is the Neural Fingerprint operation. We used RDKit python package to preprocess chemical molecules for fingerprint operation.

Kinases are processed into two different representations: kinase domain feature vectors, and kinase active site feature vectors. We denote the features of kinase domain sequence and kinase binding site sequence in the i^{th} sample as k_i^d and k_i^b, respectively, where both k_i^d and k_i^b are sequence of 27 values for each amino acid (20 for PSSM and 7 for amino acid physical properties). For each kinase, we downloaded kinase domain sequences from UniProt, and we calculated the position-specific scoring matrix (PSSM) against UniRef50 database (nonredundant sequence database clustered at 50% sequence identity) using PSIBLAST standalone package with 3 iterations for each kinase domain sequence. In the kinase domain feature representation, we used the PSSM values for the whole domain as k^d.

In the kinase active site feature representation, we first aligned the kinase domain sequences using CLUSTALW web server with default option. Then, we identified the active site residues of the serine/threonine-protein kinase pim-1 (P11309) from its 3D structure. From the 3D structure, we also collected neighboring triplets, three amino acid residues that are closer than 6.0 Å with each other. Then, we identified the binding site residues and structural neighbors of each kinase from the multiple sequence alignment. The corresponding PSSM values for the binding site residues were used as the input, k^b, in contrast to the kinase domain representation. For both representations, we also used various types of physical properties of amino acids: average hydrophobicity (Cid et al, 1992. PMID: 1518784), van der Waals volume (Fauchere et al. 1988. PMID:3209351), polarity

(Grantham et al. 1974. PMID:4843792), net charge (Klein et al., 1984. PMID:6547351), average volume in buried state (Chothia, 1975. PMID:1118010), accessible surface area in tripeptide, and accessible surface area in folded protein (Chothia, 1976. PMID:994183). The PSSM and amino acid properties were normalized by z-scaling, i.e. $z = \frac{x-\mu}{\sigma}$, where x is the feature value, and μ, σ are the mean and standard deviation of the features of same type.

The kinase feature vectors were obtained by applying graph convolutional network with attention mechanism. Many recent graph neural network methods use message passing scheme with attention mechanism as an aggregator of the node weights. Our method uses a cardinality preserved attention network (CPAN), a novel graph convolutional method developed by us (manuscript under review). CPAN takes graph representation of kinases as input, and it produces protein feature of length 154. Each kinase is a graph, where each amino acid is a node with its feature (k_i^d or k_i^b). We denote the kinase feature calculation as $f_{k_i} = \mathcal{G}_k\left(k_i^d\right)\ or\ \mathcal{G}_k(k_i^b)$, where f_{k_i} is the protein feature output from CPAN ($f_{k_i} \in \mathbb{R}^{154}$), and $\mathcal{G}_k$ is the graph convolutional operation by CPAN.

2. Attentive pooling and feature transformation

With the feature representation of chemicals and kinases, we applied attentive pooling strategy similar to (dos Santos et al, 2016, arXiv) to weigh the contributions of each feature and rescale the chemical and kinase feature vectors.

$$f'_{c_i} = f_{c_i} \odot softmax\left(f_{k_i} \cdot \Theta\right)$$
$$f'_{k_i} = f_{k_i} \odot softmax(f_{c_i} \cdot \Theta^T)$$

In the above equations, f'_{c_i}, f'_{k_i} represent the rescaled feature vectors of chemical and kinase in the i^{th} sample, Θ represents the attentive weight matrix ($\theta \in \mathbb{R}^{128\times154}$), $\odot$ represents element-wise product (Hadamard product), and $\cdot$ represents dot product of matrices.

The rescaled feature vectors were fed into a feature transformation layer to make final prediction. We denote the feature transformations as $f''_{c_i} = \mathcal{T}_{\mathrm{c}}\left(f'_{c_i}\right)$ and $f''_{k_i} = \mathcal{T}_{\mathrm{k}}\left(f'_{k_i}\right)$, where f''_{c_i} and f''_{k_i} represent transformed chemical and kinase features, respectively. The transformation layer contains four layers of fully-connected 64 neurons activated by ReLU after batch normalization. The fifth layer contains fully-connected 64 neurons with batch normalization but without ReLU activation. The predicted pKd activity value was calculated by the dot product of the transformed feature vectors. $\hat{y}_i = f''_{c_i} \cdot f''_{k_i}$, where $\hat{y}$ denotes the predicted pKd value.

3. Training procedure

We used Adam optimizer with cosine annealing for learning rate adjustment. The initial learning rate was set to 1e-4, and batch size was set to 128. We also applied balanced regression strategy to ensure the diversity of training samples. We split the training samples into 16 bins by the pKd values, where the first bin contains samples with pKd ≤ 4.0, and the last bin contains samples with pKd > 10. The other bins were evenly spaced by an increment of 0.4. Then, we chose 8 samples from each bin to form a minibatch of 128 samples. The model was trained for 100 epochs by minimizing MSE (i.e. $\Sigma(y - \hat{y})^2$) with early stopping strategy based on the Pearson's correlation coefficient measured on the dev set. From the dev set, we randomly picked 128 samples and measured the Pearson's

correlation coefficient. We repeated it 10 times and averaged for each epoch to measure the dev performance. Our final model was chosen by the training epoch where the dev performance was highest.

Below are the docker commands to run our model for Round 2 prediction.

```
$docker run docker.synapse.org/syn17037396/graphdti-v1 > round2-1.csv
#Round 2. First submission. (objID 9686292)

$docker run docker.synapse.org/syn17037396/graphdti-v2 > round2-2.csv
#Round 2. Second submission. (objID 9686304)
```