ENStaDTI-pred - Ensemble Node Embedding and Stacking DTI Prediction Tool

Sita Sirisha Madugula [1] , Shreya Amalapurapu[1,4] , Serdar Bozdag [1,2,3]*

[1] *Department of Computer Science and Engineering, University of North Texas, Denton, TX*

[2] *Department of Mathematics, University of North Texas, Denton, TX*

[3] *BioDiscovery Institute, University of North Texas, Denton, TX*

[4] *Texas Academy of Mathematics, University of North Texas, Denton, TX*

Corresponding Author E-mail: serdar.bozdag@unt.edu

Abstract

Drug Target Interactions (DTI) are vital in identifying promising lead drug-like molecules that can be taken forward to clinical studies. Traditionally, lead molecules are identified through experimental testing involving expensive, time-consuming, and laborious procedures. Computational DTI predictions using Machine Learning (ML) techniques can substantially reduce the cost involved by narrowing down the search space and finding new targets for existing drugs. In the current study, we applied network-based techniques in combination with ML algorithms on the multiplex, heterogeneous biological networks to identify newer DTIs to repurpose approved drugs of DrugBank. The pipeline has three steps: creating a graph containing known DTIs and representing the similarity amongst drugs and targets, generating lower dimensional drug and target node embeddings, and machine learning on the obtained embeddings for DTI predictions.The multiplex-heterogeneous network is constructed using eleven different types of drug-drugs, target-target similarity matrices, and known drug-target bipartite associations collected from different public domain databases. Node embeddings are computed using a unique hybrid approach where five types of embedding algorithms GATNE, ComplEx, MultiVERSE, AspEm, and Hin2Vec are applied individually on the multiplex heterogeneous network to learn lower dimensional vector representations of drugs and targets. The feature vectors from the three algorithms are then individually used in downstream analysis using a stacking machine learning classifier. Following this, the outputs of all of the embeddings are concatenated and are thereafter inputted into another stacking classifier. We observed improved performance of the multi-layer stacked approach as compared to the individual usage of the embeddings. Among the DTIs obtained in the study, we evaluated 6 novel protein-ligand interactions at the molecular level, by carrying out an in-silico molecular docking analysis. The pipeline predicted numerous DTIs which can potentially serve as a starting point for developing drug-repurposing-based treatment regimens for a multitude of diseases.

Keywords: Drug repurposing, graph representation, systems biology, drug-target interactions

1. Introduction

Global pandemics and epidemics are becoming increasingly prominent because of recent globalization [1,2]. Moreover, the evolution of drug-resistant strains of these microbes is a major concern [3]. An estimated 2.8 million antibiotic-resistant infections occur each year causing up to 35,000 deaths annually in USA[]. Additionally, there are approximately 7,000 orphan diseases listed by NIH with FDA-approved treatment options available for only 838 of these diseases []. Therefore, there is an urgent need to identify new treatment options. Traditional drug discovery processes are highly risky and take up to 13.5 years and $1.8 billion. These long and expensive processes along with stringent regulatory guidelines for FDA approvals appear discouraging for pharmaceutical companies to invest in designing effective treatment options for many diseases [4]. Drug Repurposing (DR) can serve as a good alternative to identify new treatments in a short time with little cost. DR is the process of identifying new therapeutic uses for approved drugs that are already used for various disease conditions [5,6]. Besides, owing to the low rate of NDA approvals by FDA, drug repurposing is a potentially viable option to find alternative treatments in a short time with little cost and resources.

Furthermore, the high costs, low success rates, and time-consuming nature of in-vivo/in-vitro drug discovery serve as reasons for the increased use of low-cost and much faster in-silico methods that can significantly decrease the time and cost associated with drug discovery, repurposing, and development [7,8,12,13]. Computational DR is one of the major steps in drug discovery processes and can contribute significantly to accelerating drug development processes. The primary step for identifying drug repurposing possibilities is to identify new ways that drugs and targets can interact using already known DTIs [9, 10]; however, this presents a problem, since there is an extremely limited number of known DTIs. Therefore, DTI prediction is a vital step in drug repurposing processes. Both structure-based and ligand-based techniques are currently employed as computational DR strategies. However, these techniques are limited in the absence of protein crystal structures or when the ligand's chemical space is out of the application domain respectively [11].

Due to the insufficient data required to perform large-scale, in-silico drug repurposing processes using structure-based/ligand-based techniques, network-based approaches for DTI prediction are being investigated as possible solutions [14]. Among the different computational DR techniques, network-based approaches for DTI prediction are being rapidly used [15,16,17,18], due to their ease of application even in the absence of protein crystallographic structures. These techniques rely on the principle of "guilt-by-association" [19,20] where similar drugs behave similarly and interact with similar targets by a similar mechanism of action. The current study is based on this principle where new DTIs are learned based upon existing associations in the multiplex heterogeneous networks formed by drug and target similarity matrices. Network-based methods have been developed by formulating the DTI prediction problem as a link prediction problem where there is a heterogeneous graph with drug and target nodes, where the goal is to uncover a link that could connect a drug to a target [14].

Due to the ability of heterogeneous networks to integrate diverse data from multiple sources, network-based methods for DTI prediction have gained popularity in recent years. Traditionally, network based methods solve DTI link prediction problems using computational inference methods, such as bipartite local models (BLM) [21,22,23], network based interface (NBI) [24], or propensity score matching (PSM) techniques [25,26]. As an example, DDR [26] is a network analysis based DTI prediction tool that uses a heterogeneous network for its DTI prediction task. DDR is based on PSM that generates a heterogeneous network consisting of known DTIs, drug-drug similarity edges, and target-target similarity edges. Similarity selection alongside integration algorithms is first applied to select similarities subset, which are then fused using a nonlinear function and path scores are generated for each path category which are then used to generate feature vectors. These feature vectors are then fed into a random forest classifier for prediction. However, network analysis-based methods like PSM have limitations that make it difficult to handle the sparsity and high dimensionality of heterogeneous DTI networks, resulting in low model performance [27]. These limitations are addressed by network-embedding methods [28,29] which are more efficient at solving link prediction tasks than traditional network analysis methods. Network embedding methods efficiently generate lower dimensional vector representations of the network's associations thereby preserving its topology which are then analyzed using downstream ML algorithms.. Among these, matrix factorization methods are one of the earliest implemented methods [30,31,32]. For example, DNILMF (Dual Network Integrated Logistic Matrix Factorization) [33] employs matrix factorization to generate network embeddings by integrating different drug- and target- similarity measures by applying a nonlinear similarity fusion technique derived from the similarity network fusion method (SNF) [34]. These embeddings are then used to predict DTIs based on their neighbors within the graph. Similarly, NRLMF (Neighborhood Regularized Logistic Matrix Factorization) [35] employs a logistic matrix factorization approach to model the probability that a drug would interact with a target. NRLMF integrated logistic matrix factorization with neighborhood regularization for DTI prediction. Firstly, both drugs and targets are mapped into a shared latent space and then the drug-target combinations are represented using the linear combinations of the drug-specific and target-specific latent vectors. These linear combinations are then used for prediction. Other network embedding methods include knowledge graphs. As an example, TriModel [36] utilized a knowledge graph constructed from the integration of different information sources to generate knowledge-graph embeddings for entities and relations. TriModel then predicted DTIs based on scores generated using trained tensor factorization which was applied on the knowledge-graph embeddings. Despite the drawbacks associated with path-score-based methods, another path-score based DTI prediction tool which addressed the shortcomings of traditional path-based methods is DTiGEMS+ [37]. This tool integrated topological information with multiple drug- and target- similarities to use for DTI prediction; therefore, outperforming all aforementioned models. DTiGEMS+ predicts DTIs using graph embedding and similarity-based techniques. First, DTiGEMS+ integrated multiple drug-drug and target-target similarities using the similarity network fusion algorithm and a similarity selection procedure. This was used to generate a heterogeneous graph that augmented known DTIs with the drug-drug subgraph and the target-target subgraph. DTiGEMS+ then calculated the path score features from the graph and fed them into a machine learning classifier for final predictions. However, like other DTI prediction methods, DTiGEMS+ alongside all previously aforementioned methods suffer from high-false-positive rates. This limitation was mitigated by DTi2Vec [38] which used ensemble learning techniques and achieved higher prediction performance. DTi2Vec employed biased random walk using node2vec to generate graph embeddings for each node, which were then combined using several fusion functions for drug-target edge embedding generation. The graph used to train these graph embeddings was a heterogeneous network developed by augmenting the DTI graph with a k-nearest neighbor drugs subgraph and a k-nearest neighbor targets subgraph. The drug-target combinations were then predicted using various boosting classifiers.

Current methods designed for DTI prediction continue to suffer from low performance on datasets other than the Yamanishi Benchmark Datasets, resulting in the need for an improved method for DTI prediction. Most current methods do not integrate most biological/chemical data available pertaining to drugs and targets. Furthermore, most current methods do not employ ensemble graph embeddings, which have been shown to improve predictive performance in link prediction tasks [39,40]. Therefore, in this study, we propose a novel DTI prediction framework that employs stacking-based ensemble machine learning techniques to merge learned graph representations extracted from five node embedding frameworks. These node embeddings were trained on a heterogeneous multiplex network consisting of eleven different edge types integrating biological/chemical data pertaining to drugs and targets. By doing so, our framework evaluates drug-target combinations more effectively than previous models. Figure 1 provides an overview of our pipeline.

1. Materials and Methods

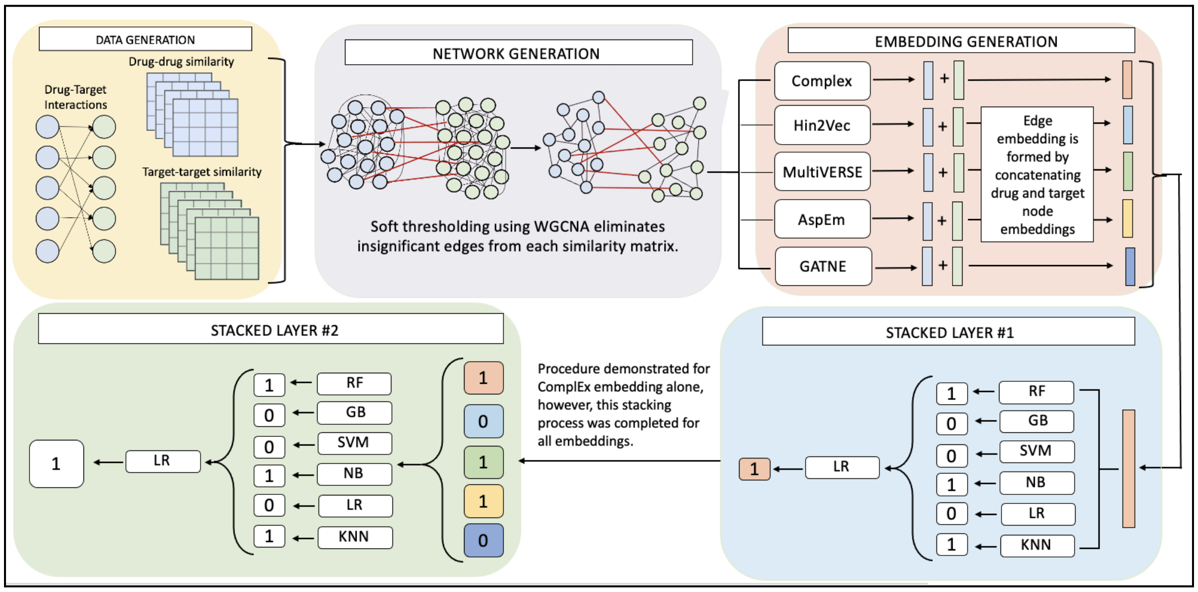
1.1 Workflow of the model

The model is implemented in six steps.

1. Generating the drug similarity and target similarity networks based on 10 different metrics.
2. Constructing a multiplex, heterogeneous network G (V, E) by augmenting the DTI graph with the drug-drug subgraph and the target-target subgraph.
3. Learning lower dimensional vector representations of the nodes within the graph using five node embedding frameworks (GATNE [41], ComplEx [42], MultiVERSE [43], Hin2Vec [44], AspEm [45]) and generating edge embeddings by concatenating the target node embedding to the drug node embedding.
4. Individually employing a stacking ensemble classifier on the vector representations learned from each node embedding framework.
5. Concatenating the outputs of each stacking ensemble classifier per node embedding framework.
6. Classifying the data samples using a stacking ensemble classifier on the concatenated outputs

Figure 1. ENStaDTI-pred Methodology Overview

j


DATA GENERATION
Drug-drug similarity
Drug-Target Interactions
Target-target similarity
NETWORK GENERATION
Soft thresholding using WGCNA eliminates insignificant edges from each similarity matrix.
EMBEDDING GENERATION
Complex
Hin2Vec
MultiVERSE
AspEm
GATNE
+
Edge embedding is formed by concatenating drug and target node embeddings
STACKED LAYER #2
STACKED LAYER #1
RF
GB
SVM
NB
LR
KNN
1
0
Procedure demonstrated for ComplEx embedding alone, however, this stacking process was completed for all embeddings.

1.1 Data sources

1.1.1 Drug-Target Interactions

The known drug-target interactions are collected from DrugBank () and Therapeutic Target Database (). The drug-target interactions evaluated in this study were restricted to drug-target interactions involving approved drugs as classified by DrugBank. Our study evaluates 2,601 approved drugs and 2,721 protein targets.

1.1.2 Drug-Drug Link Generation

Information pertaining to drugs was collected on four different metrics: chemical formula, drug-drug interactions, Anatomic Therapeutic Codes (ATC), and side-effect profiles. Drug chemical formula data was extracted from DrugBank in the form of SMILES. (Wishart et. Al, 2017) Drug side-effect profiles were extracted from the SIDER database (Kuhn et. Al, 2015) and from the Offsides database (Tatonetti et. Al, 2012). Drug ATC profiles and drug-drug interactions were extracted from the DrugBank database. (Wishart et. Al, 2017).

1.1.3 Target-Target Link Generation

Information pertaining to targets was collected on three different metrics: protein sequence, protein-protein interactions, and GO annotations. Protein sequence information was downloaded in FASTA format from the DrugBank database. (Wishart et. Al, 2017) GO annotations were extracted from the Gene Ontology database. (Ashbumer et. Al, 2000) GO annotations were extracted in three different categories: Molecular function, biological process, and cellular component.

1.2 Benchmark Datasets

1.2.1 Yamanishi Benchmark Datasets

In order to compare the performance of our model, we evaluated our model on the Yamanishi Benchmark Datasets. The Yamanishi_08 datasets consist of four "gold-standard" datasets. The four datasets within the Yamanishi Benchmark Datasets span over the different families of target proteins (http://web.kuicr.kyoto-u.ac.jp/supp/yoshi/drugtarget/), including G protein-coupled receptors (GPCR), ion channels (IC), nuclear receptors (NR), and enzymes (E). Yamanishi et al. created the Yamanishi Benchmark datasets by collecting DTIs from reliable sources including KEGG BRITE, BRENDA, SuperTarget, and DrugBank (the information was collected from the databases released in 2008).

1.2.2 DrugBank Benchmark Datasets

In order to guage the performance of our model as compared to other models, we evaluated the performance of our model on the DrugBank hold-out set, which was developed in the DTI2Vec paper. This dataset was developed using the 5.0.3 version release of DrugBank and only includes the DTIs with Food and Drug Administration (FDA) approval (https://www.drugbank.ca).

1.2 Generating Similarity Matrices

1.2.1 Generating Drug-Drug Similarity Matrices

The similarity between drug SMILES data was computed using RDKit [], which is a python library for chemoinformatics. (RDKit, 2021) Morgan 1024-bit fingerprints were generated for each SMILES and then similarity was computed using the Tanimoto Coefficient (1).

$$TC(d_1, d_2) = \frac{|d_1 \cap d_2|}{|d_1| + |d_2| + |d_1 \cap d_2|}$$

The similarity between side-effects was computed using Jaccard Similarity (2).

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|}$$

Drug Anatomic-Therapeutic Codes based similarity was computed using a level-wise Jaccard similarity. Most drugs contain one or more ATC codes. ATC codes are composed of five distinct levels, each providing additional therapeutic, chemical, or pharmacological properties of the drug. The first level indicates the main anatomical group, the second level indicates the therapeutic subgroup, the third level indicates the therapeutic/pharmacological subgroup, the fourth level indicates the chemical/therapeutic/pharmacological subgroup, and the fifth level indicates the chemical substance. ATC profiles were generated for each drug, with one-hot encoded vectors being allotted for each level of the ATC code. Jaccard similarity was computed for each level using the encoded vectors and then averaged to get the final similarity score.

Similarity between drug-drug interaction profiles was computed using a weighted Jaccard similarity.

$$J(A, B) = \frac{|A \cap B|}{|A \cup B|} = \frac{\frac{|A \cap B|}{\#\text{ entities A}} + \frac{|A \cap B|}{\#\text{ entities B}}}{|A \cup B|}$$

1.2.2 Generating Target-Target Similarity Matrices

The amino-Acid composition was computed using the protein-sequence data and using the iFeature python package. (Chen, Zhen, et. Al, 2018) The similarity between calculated amino-acid composition feature vectors was computed using cosine similarity. (4)

$$\cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}}$$

The protein-sequence similarity was computed using Smith-Waterman [] protein sequence alignment, implemented through BioPython.

Similarity amongst GO annotations was computed using semantic similarity. The semantic similarity was computed using Wang et. Al (2018) approach to GO similarity calculation, implemented through the GOSemSim R package. (Yu, Guangchuang et. Al, 2010)

1.3 Edge-weight thresholding using WGCNA soft thresholding

Soft thresholding was then performed on all the similarity matrices using the WGCNA R package. Soft thresholding refers to the process of exponentiating the correlation matrix, which helps accentuate more significant edges/connections. Following the soft thresholding, a hard threshold was applied onto the adjacency matrices in order to further eliminate redundant edges (Table 1). Protein-protein interactions were not subjected to soft-thresholding because of their limited number and because of their lack of similarity index.

Table 1. Detailed summary depicting the number of edges in the network prior to and following soft thresholding.

Similarity matrix type	Soft threshold cut-off used in WGCNA	No. of edges after soft thresholding	Hard cut-off value	No. of edges left after hard threshold
Protein Sequence	0.85	7,257,636	0.1	2,589,822
Protein GO (Molecular Fingerprint)	0.85	3,822,025	0.001	627,679
Protein GO (Biological Process)	0.85	3,814,209	0.001	80,544
Protein GO (Cellular Component)	0.85	3,790,809	0.001	376,203
Protein Amino Acid Composition	0.85	9,388,096	0.001	759,180
Drug Side-Effect	0.85	988,036	0.001	43,222
Drug ATC (Anatomic Therapeutic Codes)	0.60	3,992,004	0.001	210,778
Drug-Drug Interactions	0.45	7,257,636	0.001	56,691
Drug Molecular Fingerprints	0.85	6,250,000	0.001	200,157

1.4 Generating the Graph

In this study, we defined a multiplex heterogeneous graph that represents the DTI network combined with a drug-drug similarity graph and a target-target similarity graph. The graph G(V, E) consists of the set of all approved drugs as taken from DrugBank, D = $\{d_1, d_2, ..., d_3\}$ of n drug nodes and the set of all known protein-targets T = $\{t_1, t_2, ..., t_3\}$ of m target nodes. The graph G contains eleven different edge types. The first edge type represents drug-target interactions and are bipartite connections. Four of the remaining edge types represent the similarity between drugs and the remaining six edge types represent the similarity between targets. All of the edge weights within the network were assigned to a value of 1. Given graph G, this study aims to predict new bipartite relations amongst drug nodes and target nodes, therefore, setting up a link prediction problem.

1.4 Generating Node Embeddings

Node embeddings were generated using five different node embedding frameworks: GATNE, ComplEx, MultiVERSE, Hin2Vec, and AspEm.

The GATNE embeddings had a dimension size of 2000, the Complex embeddings had a dimension of 200, the MultiVERSE embeddings had a dimension of 135, the Hin2Vec embeddings had a dimension of 500, and the AspEm embeddings had a dimension of 200. Edge embeddings were created by concatenating the node embedding of the drug node to the node embedding of the target node. Due to the difference in embedding sizes amongst all algorithms, we employed late-integration-based data ensembling techniques in order to integrate the graph representations from all methods.

We constructed our train/validation/test data using a subset of reliable DTIs (as extracted from DrugBank and Therapeutic Target Database) as our positive samples and randomly sampled drug-target pairs to act as negative samples. Due to the limited number of experimentally-validated negative DTIs available for most drugs and targets, random pairing was done to better represent the negative DTIs since the ratio of existing DTIs is very small as compared to the total number of drug-target pairs.

1.5 DTIs Predictive Model Generation

1.5.2 Sampling for imbalanced data

Random sampling was employed in order to balance the data to deal with the number of unknown DTIs being much larger than the number of DTIs. We employed random sampling on the training data in order to balance the minority and majority classes. We did 1:1 random sampling, so we had the same number of positive and negative samples.

1.5.2 Stacking-based Classifier

In this study, we propose a novel late-integration data ensembling technique that merges five different graph representations (extracted from five different node embedding frameworks) using a three-layer stacking machine learning classifier. The first stacked layer is created by stacking a logistic regression classifier on top of six different binary classifier models for the embeddings extracted from each embedding framework individually. The second stacked layer is created when the outputs of the initial stacked layer are extracted for each node embedding framework and concatenated to form the input to another machine learning classifier. The third stacked layer is created when a logistic regression classifier is used as a meta learner atop six other binary classification models and is trained on the inputs generated in the previous stacked layer. Using this multi-layer stacking ensembling technique, we effectively combined the information of all five node embedding frameworks with minimal bias due to their different dimension sizes, while simultaneously producing high performance on various benchmark datasets.

The stacking-based ensembling technique uses six different machine learning classifiers including Random Forest (RF), Naive Bayes (NB), Support Vector Machine (SVM), and K-Nearest Neighbors (KNN), Gradient Boosting (GB), and Logistic Regression (LR). Our stacking model took the outputs of the six machine learning models above as sub-model outputs and then merged them to create inputs for a second stacked classifier. The weak learners consisted of an RF, KNN, NB, GB, SVM, and LR, and the meta learner was an LR model. Hyperparameter tuning via GridSearchCV was conducted for each algorithm listed above in order to develop the most effective classifier based on the area under the precision-recall curve. The outputs from each ML classifier were then concatenated to form a vector of size six, which was then fed as input into the meta learner, which outputted a single classification label.

These classification labels were then carried on to the next stage of stacking, where the outputs of the previous stacking classifiers (for each node embedding framework (GATNE, MultiVERSE, ComplEx, Hin2Vec, AspEm) were concatenated to form an input to the last stacked layer in the model structure.

The series of five inputs were fed into another stacked classifier that employed RF, NB, SVM, KNN, GB, and LR as weak learners and LR as the meta-learner. The hyperparameters for all of these models were tuned using GridSearchCV with a cross-validation score of 10. After the third stacked layer was run, the outputted prediction scores were carried forward for further analysis.

1.6 Evaluation methods

1.6.1 Evaluation metrics

To evaluate the prediction accuracy, the area under the receiver operating characteristic (ROC) curve (AUC), as well as the area under the precision-recall curve (AUC-PR) are calculated on the testing data. To determine the AUC and AUC-PR, we calculated the false positive rate (FPR), true positive rate (TPR), and precision (positive predictive value).

The ROC curve is constructed using different recall, and FPR values of different thresholds, to calculate the AUC. AUC-PR is calculated based on different precision and recall values at different cut-offs that are used to construct the curve, and then the area under this curve is calculated. The closer the value of AUC and AUC-PR are to 1, the better the performance is. For highly imbalanced data, such as the data used in this study, the AUC is considered an over-optimistic evaluation metric for the prediction of DTIs, while the AUC-PR is thought to provide better assessment in such imbalanced data cases, because it separates the predicted scores of true interactions from the predicted scores of unknown interaction. Due to this, we use the AUC-PR as the significant evaluation metric and for comparison with other state-of-the-art methods.

1.7 Experimental Settings

In order to gauge the prediction performance of our model, we performed tenfold cross-validation (CV) on each benchmark dataset separately and then the datasets combined. The data was randomly partitioned into ten subsets in a stratified fashion where each subset included an equal number of positive to negative samples. We kept aside 1 subset of the data for testing, and the remaining nine subsets were used for training the multi-layer stacking model. This process was repeated 10 separate times to have each subset of the data be the test data at some point. The AUC-PR and ROC-AUC were calculated for each fold and then the averages were saved in the tables reported in the paper.

While evaluating our model on the various benchmark datasets, we eliminated drug-target interactions that were already used while training the graph embeddings that were used throughout our pipeline. This vital step ensured that no information that was already preserved in our graph embeddings was used to evaluate the performance of our model.

2. Results and Discussion

Our results contain a detailed depiction of the various experiments conducted throughout the process of developing our pipeline. Additionally, our results consist of a detailed comparison of our DTI prediction performance as compared to other state-of-the-art methods. We also further evaluate the false-positives that our model predicted on the DrugBank hold out set in order to demonstrate the efficacy with which our model predicts true positives. We did this using molecular-docking studies, which provide further structural information regarding the interaction between a drug and a target-protein. Lastly, we highlighted several possible characteristics of our model that could be the leading factors behind our increased prediction performance as compared to other methods.

2.1 Comparing the use of Dot Products to the use of concatenation for edge embedding representation

In order to evaluate the most effective way of generating edge representations, we investigated two different methods. The first method employed the dot product of the drug embedding vector and the target embedding vector. The second method employed concatenation of the drug embedding vector to the target embedding vector in order to represent the drug-target edge. Table 2 details the performance of the dot product method as compared to the concatenation method on each of the node embedding frameworks individually.

Table 2. Summary showing the performance of using the dot product of the two node embeddings vs. using the concatenation of the two node embeddings for creating edge representations. Yellow highlight indicates concatenation method.

	GATNE	GATNE	Complex	Complex	Multiverse	Multiverse	AspEm	AspEm	Hin2Vec	Hin2Vec
Accuracy	0.8444	0.5156	0.7395	0.5118	0.7325	0.4863	0.7416	0.4844	0.7881	0.4193
ROC-AUC	0.8487	0.5000	0.7483	0.4980	0.7415	0.4922	0.7503	0.5000	0.7940	0.4295
F1-Score	0.8366	0.6804	0.6941	0.6649	0.6839	0.3762	0.6966	0.000	0.7679	0.1544
AUC-PR	0.9079	0.7578	0.8472	0.7426	0.8423	0.5820	0.8491	0.7578	0.8716	0.4376

2.2 Comparing Model Ensembling to Individual Model Performance

In order to construct the multi-layer stacking ensemble classifier, we first had to extract the predictions from the first-layer models, which were models that were individually trained on the embeddings generated by each individual node embedding framework. A stacking classifier was developed using RF, SVM, LR, NB, GB, and KNN as weak learners and LR as the meta learner. Evaluation was conducted using five-fold cross-validation (CV) and hyperparameter tuning using grid search was employed on each individual machine learning classifier prior to using them as weak learners for the stacking ensemble approach. As can be seen in Figure 2, the stacking classifier outperformed each individual machine-learning algorithm when trained on the edge embeddings for every node embedding framework. Table 1 details the results on four different scoring metrics (accuracy, ROC-AUC, f1-score, AUC-PR) of the stacking ensemble machine learning model on the embeddings extracted from each of the five different node embedding frameworks averaged over the five CV runs.

Figure 2. Performance comparison of the proposed stacking method with the six ML classifiers across all five embedding frameworks considered in the study

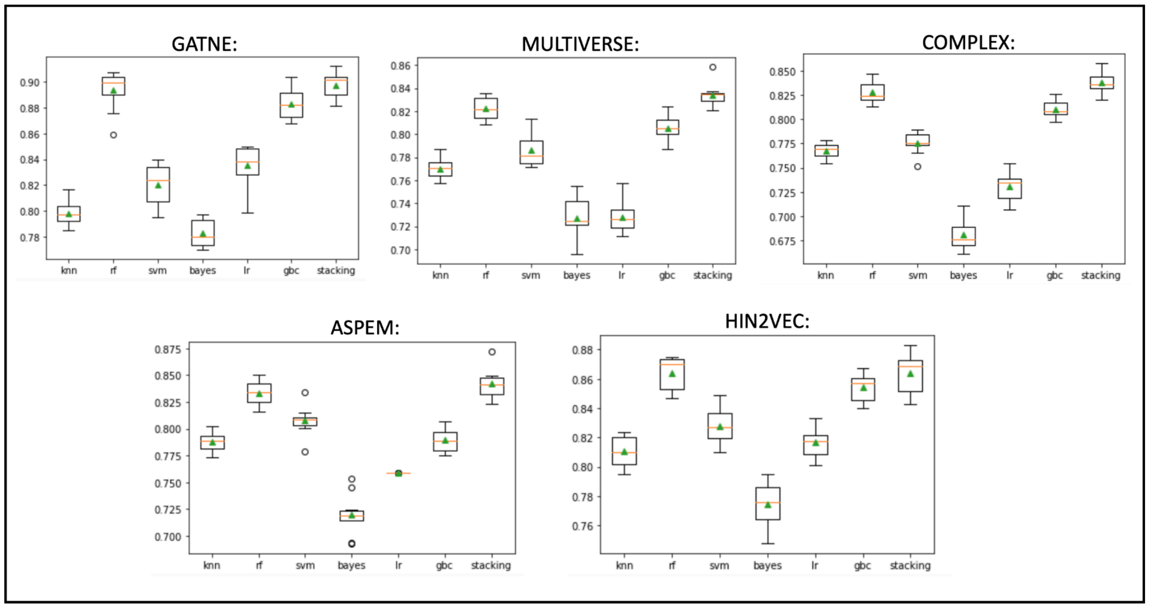


Table 3. Summary showing the performance of our stacking ensemble model on the embeddings extracted from each of the five node embedding frameworks considered in the study.

	AspEm	Hin2Vec	ComplEx	GATNE	MultiVERSE
Accuracy	0.7416	0.7881	0.7395	0.8444	0.7325
ROC-AUC	0.7503	0.7940	0.7483	0.8487	0.7415
F1-Score	0.6966	0.7679	0.6941	0.8366	0.6839
AUC-PR	0.8491	0.8716	0.8472	0.9079	0.8423

2.2 Comparing the performance of the layered stacking approach as compared to other ensembling techniques

After extracting the predictions from the first stacked layer, where a stacking classifier was trained on the embeddings from the five node embedding algorithms individually, the predictions were concatenated together. The concatenated output was a vector of size five that consisted of the five predictions from the five stacked classifiers. In order to determine how best to use these predictions to come to a final prediction, we evaluated two different ensembling techniques: bagging and stacking. To construct the stacking classifier, we used the concatenated output vector to train RF, LR, KNN, GB, and SVM machine learning models. We also experimented with a multilayered stacking approach, where we trained another stacking classifier (weak learners: RF, LR, KNN, GB, SVM, NB; meta learner: LR) where the concatenated output vectors were used as input. Table 4 provides a comparison of these different ensembling techniques and demonstrates that another stacking model performs the best atop the second stacked layer.

Table 4. Performance comparison of the proposed method compared to other ensembling techniques

	Bagging	Stacking (RF)	Stacking (LR)	Stacking (KNN)	Stacking (GB)	Stacking (SVM)	<u>Stacking (Stacking)</u>
Accuracy	0.7874	0.8374	0.8374	0.8211	0.8374	0.8374	<u>0.8621</u>
ROC-AUC	0.7926	0.8423	0.8420	0.8246	0.8420	0.8420	<u>0.8721</u>
F1-Score	0.7646	0.8316	0.8322	0.8184	0.8323	0.8323	<u>0.8657</u>
AUC-PR	0.8718	0.9031	0.9020	0.8855	0.9020	0.9020	<u>0.9321</u>

This experiment provides further evidence behind the use of the multi-layer stacked ensemble technique, since this technique significantly outperformed not only individual node embedding frameworks run on our data, but also other ensembling techniques.

2.3 Comparing Stacking Techniques to other Early Integration/Late Integration Techniques

To explore additional techniques that can boost the prediction performance of our proposed method, we tested our stacking model and compared it to different types of early and late integration techniques which are commonly employed in deep learning pipelines. At this stage we used three strategies to boost the performance: (a) Early integration by feeding the outputs from other embedding frameworks as node features to GATNE (b) Dimensionality reduction on the embedding space by using autoencoder followed by early integration for every embedding algorithm (c) early integration without any dimensionality reduction (d) early integration with autoencoder on the concatenated embedding vector from all five algorithms (e) late integration bagging. A summary of the five strategies and the performance of our model using these strategies is given in table 5.

Table 5. Table depicting the five boosting strategies used in the study and the performance of the proposed method using these techniques.

	Early Integration (feeding outputs from other embedding frameworks as node features to GATNE)	Early Integration without dimension reduction	Early Integration with autoencoder (dimension reduction) for each embedding	Early Integration with autoencoder on concatenated embedding	Late Integration Bagging
Accuracy	0.746	0.85	0.80	0.80	0.78
ROC AUC	0.502	0.85	0.80	0.81	0.79
F1 Score	0.692	0.84	0.80	0.78	0.75
AUC-PR	0.701	0.90	0.87	0.88	0.87

As seen from table 5, amongst the different early integration strategies applied in this study, strategy (b); early integration without applying any dimensionality reduction has substantially outperformed the other early integration strategies in terms of its AUC-PR score. It is also to be noted that we tried strategy (a) using only GATNE since it is the only one among the five considered embedding algorithms that is built to accept node features. The remaining four algorithms did not have a provision to integrate node features while calculating the node embeddings. However, the technique does not perform as well as the other integration strategies which could be due to the high number of features extracted from all the embedding methods. Compared to strategy (a), the idea of dimensionality reduction of the embedding vector before early integration produced an improved performance across all evaluation metrices compared to using them as node features. However, feeding a concatenated feature vector from all the embedding algorithms into the autoencoder for dimensionality reduction produced comparable results to that using the dimensionality reduction on each embedding algorithm. This method is only second to the best performing strategy (b). This improved performance of strategy (b) over the other strategies can be attributed to the fact that there is no loss of information while integrating the raw feature vector as obtained from the embedding algorithm. On the other hand, applying dimensionality reduction can still cause some loss of information while preserving the variance explained by the latent feature set. This also proves that although different embedding algorithms gave embedding vector of different size, the features are meaningful and does not include any redundancy. Therefore strategy (b) performed the best compared to the other techniques including the late integration. However, the novel stacking late-ensembling technique we developed significantly outperformed all methods evaluated in table 5.

2.2 Further analysis of the false positives predicted

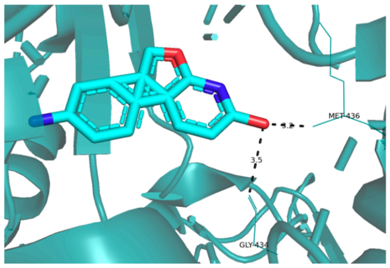
In order to further demonstrate the efficiency of our model in predicting the true positive cases of DTIs, we further assessed the false positive predictions (samples initially labeled as negative, but predicted to be positive) our model made. Since all of the negative samples were randomly sampled, we wanted to look deeper to see if there was potential for interaction in our predicted false-positives. We conducted this analysis using molecular docking through AutoDock.

Table 6: Table depicting the specifics of the docking process, as well as, the binding energy output.

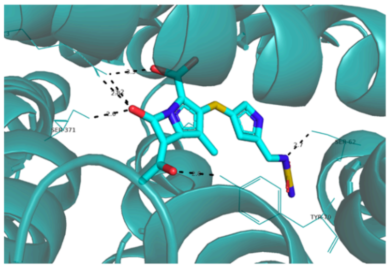
	Drug	Target	Grid Dimensions	Center	Binding Energy (kcal/mol)
DTI1	DB00357	P27338	20 x 30 x 20	57 x 145 x 20	-8.3
DTI2	DB06211	P53985	20 x 20 x 20	110 x 112 x 107	-8.7
DTI3	DB04817	P10632	20 x 20 x 20	49 x 16 x - 24	-7.1
DTI4	DB06144	Q13946	20 x 20 x 20	-0.4 x 52 x 22	-9.6
DTI6	DB09225	P32320	20 x 20 x 20	32 x 93 x 106	-6.6
DTI8	DB11274	P19971	40 x 40 x 40	-4 x 2 x 29	-8.2

Figure 3: The drug-target interaction top complexes as extracted from AutoDock.

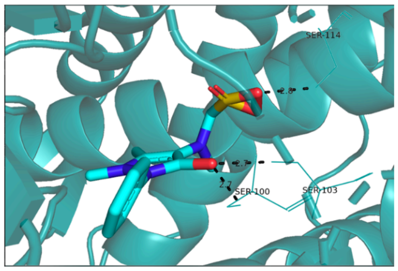
DTI1



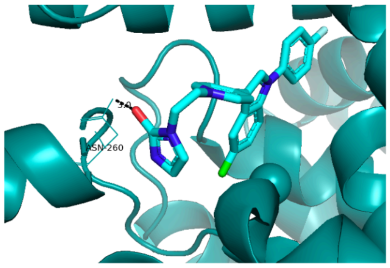
DTI2



DTI3

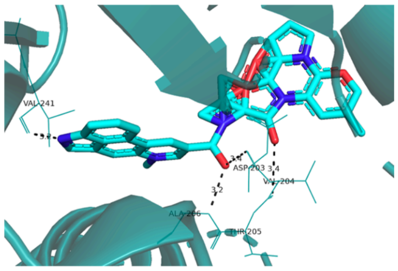


DTI4



DTI6
Keep both chains

DTI8



2.3 DTI prediction performance of our model

To evaluate our method, we compared the DTI prediction performance of our model and six other models on the Yamanishi_08 Benchmark datasets and on the DrugBank hold-out set developed in the paper Dti2Vec. The state-of-the-art methods against which we compared our model's predictive performance include NRLMF, DNILMF, DDR, TriModel, DtiGEMS+, and Dti2Vec.

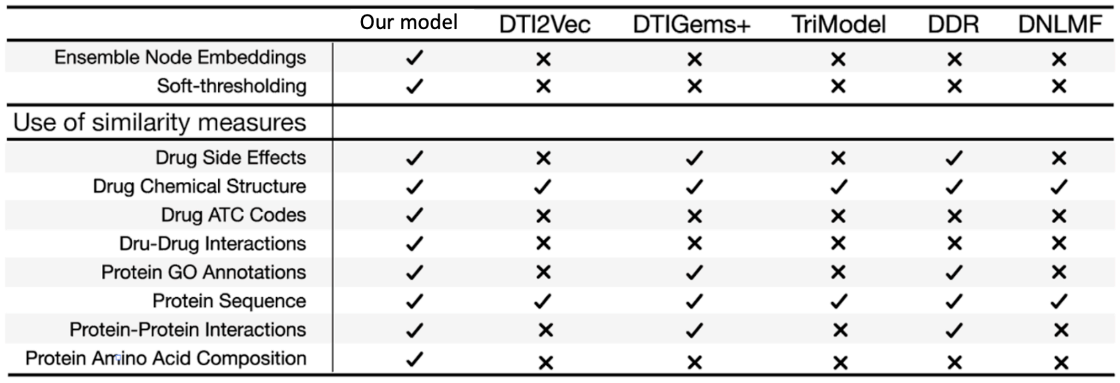
To effectively compare the performance of our model to the performance of the other models, we used the same benchmark datasets and evaluation metrics. Our method outperformed all other methods by achieving the best performance across all benchmark datasets on the AUC-PR scoring metric, which, as defined earlier, is the most valuable guage of model performance for these types of imbalance problems. Table 6 provides a comparison of our model to all the other models on the AUC-PR and the AUC scoring metrics. The scores in the table are averaged across tenfold cross-validation runs for each model.

Table 6: Performance comparison of our model with other known DTI prediction tools

Dataset	**Metric**	**Method**						
		NRLMF	**DNILMF**	**DDR**	**TriModel**	**DtiGEMS+**	**DTi2Vec**	**Our Model**
Yamanishi_08	***AvgAUPR***	0.80	0.78	0.87	0.88	0.92	0.95	0.98
	AvgAUC	0.95	0.95	0.96	0.98	0.99	1.00	0.99
All datasets Yamanishi + DB	***AvgAUPR***	0.72	0.69	0.82	0.84	0.86	0.93	0.98
	AvgAUC	0.94	0.95	0.96	0.98	0.99	0.99	0.99
FDA_DrugBank (Hold—out)	***AvgAUPR***	0.34	0.31	0.63	0.66	0.62	0.82	0.97
	AvgAUC	0.93	0.95	0.97	0.99	0.97	0.99	**0.95**

As demonstrated in Table 6, our model significantly outperformed all other state-of-the-art DTI prediction models on the AUC-PR scoring metric by up to 18% on the DrugBank holdout set benchmark dataset. This can be attributed to not only our late-integration graph-representation ensembling technique, but also, to our extensive use of biological and chemical data in our graph used to learn graph representations. Figure 4 depicts the use of various biological/chemical data across all DTI prediction papers evaluated, as well as the use of soft-thresholding for insignificant edge elimination and ensembling techniques for graph representation learning across all DTI prediction pipelines.

Figure 4:



	Our model	DTI2Vec	DTIGems+	TriModel	DDR	DNLMF
Ensemble Node Embeddings	✓	×	×	×	×	×
Soft-thresholding	✓	×	×	×	×	×
Use of similarity measures						
Drug Side Effects	✓	×	✓	×	✓	×
Drug Chemical Structure	✓	✓	✓	✓	✓	✓
Drug ATC Codes	✓	×	×	×	×	×
Dru-Drug Interactions	✓	×	×	×	×	×
Protein GO Annotations	✓	×	✓	×	✓	×
Protein Sequence	✓	✓	✓	✓	✓	✓
Protein-Protein Interactions	✓	×	✓	×	✓	×
Protein Amino Acid Composition	✓	×	×	×	×	×

3. Conclusion

Our study introduced a novel multilayer stacking ensembling technique for drug-target interaction prediction. Our model integrated graph representations from five different node embedding frameworks, as well, as computed the similarity between drugs and targets on eleven different metrics. Our model can be broken down into three primary steps: network generation (computed similarity across drugs and targets on eleven different similarity measures and performed soft thresholding to eliminate insignificant edges), node embedding generation (trained graph representations using five different node embedding frameworks and then integrated all five representations using late-integration), and creating a multi-layer stacking ensemble model (ensembled six different machine learning classifiers and five different data inputs in order to create a three-layer stacked model for effective DTI prediction). The novelty of our method lies in the node embedding generation and on the stacking late-integration ensemble techniques used to effectively combine graph representations learned through five different frameworks. There is also significant novelty in the amount of biological/chemical data used in generating the graph upon which the graph embeddings were trained. Our model proved its efficiency by outperforming six state-of-the-art methods using the AUC-PR evaluation metric. Furthermore, our study is taken a step further through additional analysis of the false positive predictions, further proving the validity behind our model's positive predictions.

For further improvements of our model, we suggest investigating similarity measures from more sources, employing additional embedding techniques, and exploring new ensembling techniques for data integration. Furthermore, we hope that eventually we can train our model using a reliable set of negative drug-target interaction samples, rather than simply randomly sampling negative samples.

4. References

[1] Aliper, A., Plis, S., Artemov, A., Ulloa, A., Mamoshina, P., & Zhavoronkov, A. (2016). *Deep learning applications for predicting pharmacological properties of drugs and drug repurposing using transcriptomic data*. **Molecular Pharmaceutics, 13**(7), 2524–2530. https://doi.org/10.1021/acs.molpharmaceut.6b00248

[2] Arora, S., Ge, R., Neyshabur, B., & Zhang, Y. (2018). *Stronger generalization bounds for deep nets via a compression approach*. **Proceedings of the 35th International Conference on Machine Learning (ICML)**. https://doi.org/10.48550/arXiv.1711.0872

[3] Barabási, A.-L., Gulbahce, N., & Loscalzo, J. (2011). *Network medicine: A network-based approach to human disease*. **Nature Reviews Genetics, 12**(1), 56–68. https://doi.org/10.1038/nrg2918

[4] Chen, H., Engkvist, O., Wang, Y., Olivecrona, M., & Blaschke, T. (2018). *The rise of deep learning in drug discovery*. **Drug Discovery Today, 23**(6), 1241–1250. https://doi.org/10.1016/j.drudis.2018.01.039

[5] Chen, J., Zhang, X., Cheng, X., & Zhang, Y. (2021). *Predicting drug–target interactions using deep learning*. **Bioinformatics, 37**(22), 4145–4153. https://doi.org/10.1093/bioinformatics/btab548

[6] Ching, T., Himmelstein, D. S., Beaulieu-Jones, B. K., Kalinin, A. A., Do, B. T., Way, G. P., … Greene, C. S. (2018). *Opportunities and obstacles for deep learning in biology and medicine*. **Journal of the Royal Society Interface, 15**(141). https://doi.org/10.1098/rsif.2017.0387

[7] D'Souza, S., Preuss, K., & Khosla, A. (2019). *Machine learning for healthcare risk prediction and classification*. **Journal of Biomedical Informatics, 95**, 103159. https://doi.org/10.1016/j.jbi.2019.103159

[8] Esteva, A., Robicquet, A., Ramsundar, B., Kuleshov, V., DePristo, M., Chou, K., … Dean, J. (2019). *A guide to deep learning in healthcare*. **Nature Medicine, 25**, 24–29. https://doi.org/10.1038/s41591-018-0316-z

[9] Gao, K. Y., Fokoue, A., Luo, H., Iyengar, A., Dey, S., & Zhang, P. (2018). *Interpretable drug–target prediction using deep neural representation*. **Bioinformatics, 34**(17), i528–i536. https://doi.org/10.1093/bioinformatics/bbt056

[10] Gilbert, D., Heiner, M., & Lehrack, S. (2019). *A unifying framework for computational modeling of biological systems*. **IEEE/ACM Transactions on Computational Biology and Bioinformatics, 16**(5), 1451–1463. https://doi.org/10.1109/TCBB.2019.2933825

[11] Gulshan, V., Peng, L., Coram, M., Stumpe, M. C., Wu, D., Narayanaswamy, A., … Webster, D. R. (2016). *Development and validation of a deep learning algorithm for detection of diabetic retinopathy*. **JAMA, 316**(22), 2402–2410. https://doi.org/10.1001/jama.2016.17216

[12] He, K., Zhang, X., Ren, S., & Sun, J. (2016). *Deep residual learning for image recognition*. **Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)**, 770–778. https://doi.org/10.1109/CVPR.2016.90

[13] King, G., & Zeng, L. (2001). *Logistic regression in rare events data*. **Political Analysis, 9**(2), 137–163. https://gking.harvard.edu/files/gking/files/psnot.pdf

[14] Kipf, T. N., & Welling, M. (2017). *Semi-supervised classification with graph convolutional networks*. **Proceedings of the International Conference on Learning Representations (ICLR)**. https://doi.org/10.48550/arXiv.1609.02907

[15] LeCun, Y., Bengio, Y., & Hinton, G. (2015). *Deep learning*. **Nature, 521**, 436–444. https://doi.org/10.1038/nature14539

[16] Litjens, G., Kooi, T., Bejnordi, B. E., Setio, A. A. A., Ciompi, F., Ghafoorian, M., … Sánchez, C. I. (2017). *A survey on deep learning in medical image analysis*. **Medical Image Analysis, 42**, 60–88. https://doi.org/10.1016/j.media.2017.07.005

[17] Miotto, R., Li, L., Kidd, B. A., & Dudley, J. T. (2016). *Deep Patient: An unsupervised representation to predict the future of patients from electronic health records*. **Scientific Reports, 6**, 26094. https://doi.org/10.1038/srep26094

[18] Ramsundar, B., Eastman, P., Walters, P., & Pande, V. (2019). *Deep learning for the life sciences*. **Academic Press**. https://doi.org/10.1016/B978-0-12-820472-6.00108-0

[19] Silver, D., Schrittwieser, J., Simonyan, K., Antonoglou, I., Huang, A., Guez, A., … Hassabis, D. (2017). *Mastering the game of Go without human knowledge*. **Nature, 550**, 354–359. https://doi.org/10.1038/nature24270

[20] Topol, E. J. (2019). *High-performance medicine: The convergence of human and artificial intelligence*. **Nature Medicine, 25**(1), 44–56. https://doi.org/10.1038/s41591-018-0300-7

[21] Vamathevan, J., Clark, D., Czodrowski, P., Dunham, I., Ferran, E., Lee, G., … Bender, A. (2019). *Applications of machine learning in drug discovery and development*. **Nature Reviews Drug Discovery, 18**, 463–477. https://doi.org/10.1038/s41573-019-0024-5

[22] Zitnik, M., Agrawal, M., & Leskovec, J. (2018). *Modeling polypharmacy side effects with graph convolutional networks*. **Bioinformatics, 34**(13), i457–i466. https://doi.org/10.1093/bioinformatics/btx731