Drug-Drug Interaction Prediction: a Purely SMILES Based Approach

Bri Bumgardner*, Farhan Tanvir¶, Khaled Mohammed Saifuddin¶, Esra Akbas¶
*Department of Computer Science, Rice University, Houston, TX, 77005, USA
¶Department of Computer Science, Oklahoma State University, Stillwater, OK, 74078, USA
*bb64.edu@rice.edu
¶{farhan.tanvir, khaled_mohammed.saifuddin, eakbas}@okstate.edu

***Abstract*—A drug-drug interaction (DDI) occurs when a drug is combined with other drug(s). DDIs have the potential to obstruct, increase, or diminish the intended impact of a drug or, in the worst-case scenario, induce an undesirable side effect. While it is critical to discover DDIs during clinical trials, it is impractical and expensive to detect all possible DDIs for a drug. Although several computational approaches for this problem have been developed, many of these methods need external biomedical knowledge that makes them difficult to generalize to drugs in early development phase. In this paper, we propose a novel method for predicting DDIs based on the vital chemical substructure of drugs extracted from their SMILES strings. We construct a graph that connects drugs based on their common functional chemical substructures. Furthermore, we apply different well-known graph neural network (GNN) methods to generate drug embeddings. Drug embeddings of individual drugs are concatenated to generate features of drug pairs. Finally, drug pair features are fed to different machine learning (ML) classifiers for DDI prediction. We evaluate our model on DrugBank dataset. Our result shows promising results and our model outperforms a baseline model based on different DDI representation creation methods.**

***Index Terms*—Drug-drug Interaction, Link Prediction, Chemical Structure, Graph Neural Network, Representation Learning**

I. Introduction

Most human diseases are caused by complicated biological processes that cannot be treated by a single drug [1], [2]. Polypharmacy, a combinatorial therapy involving concurrent administration of many drugs, is a widely used technique for fighting diseases [3]. On the other hand, drugs used together may interact with each other. Drug-drug interaction (DDI) indicates the reaction of drugs when they are taken together with another drug(s). While DDIs may hamper, enhance, or reduce the expected effect of either drug, they may also cause an adverse drug reaction (ADR). DDIs account for roughly one-third of all ADRs [4]–[6], resulting in significant morbidity and mortality globally [7]–[9]. To mitigate the impact of unexpected pharmacological effects, it is critical to effectively identify potential DDIs, which can minimize unexpected ADRs and maximize synergistic benefits when treating a disease [10].

While the search for such side effects is normally done during clinical testing, the limited nature of the trials and the rarity of ADRs imply that many ADRs are undiscovered when the pharmaceuticals reach the market. Therefore, it may not be possible to detect all possible DDIs for a new drug during the clinical trial, and many computational methods have been proposed for this task. Most extant DDI prediction approaches focus on combining several data sources to gather drug properties such as similarity features, adverse or side effects, and multi-task learning. Similarity-based methods assume that structurally similar drugs will behave similarly and interact with the same drugs [11]–[13]. A few works utilize drugs' adverse or side effect information to construct node features or build a feature set showing relation between drugs [14]. Several works consider DDI prediction task as multi-task learning that integrates task or entity relatedness during training [15], [16]. For DDI prediction, side effects can be related to each other. These works assume that if two side-effects are related and if any drug pairs cause one of the side effects, they will likely cause other side effects and further predict DDIs exploiting this relatedness. Some computational techniques choose to combine different embedding methods [17]–[20]. These embedding methods aim to automatically learn effective representations for drugs as features.

While these methods show promising results, there are still some drawbacks. Many of these methods need external biomedical knowledge like protein, disease, chemical structure, and genes, making difficult to generalize for drugs in the early development phase. Also, prior works utilized the entire chemical structure information to facilitate DDI prediction [13], [21], [22]. However, all chemical substructures are not significant, and drug pairs might overlap on irrelevant chemical substructures. Therefore, it is imperative to depict drug pairs' relations based on vital chemical substructures.

In this paper, as a solution to these problems, we propose a new method for DDI prediction where we learn the representation of drugs by considering only drugs' chemical structures and their similarities. We consider two drugs to be similar if they have similar functional sub-structure (i.e., functional groups) [23]. Instead of using whole chemical structures of drugs to measure their similarities, we use frequent substructures in them.

To appropriately model the structural similarity of drugs, we represent them in a graph setting. Graph encapsulates the intricate structure of interactions between linked objects. Vertices represent objects in a network, while edges indicate the relationships between objects. In our graph, we consider drugs as nodes and add an edge between drugs if they own common frequent chemical substructures. We utilize the Explainable Substructure Partition Fingerprint (ESPF) algorithm [24] to extract relevant, influential frequent sub-structures from drugs' molecular structures.

After creating the graph using extracted substructures, we use different graph neural network (GNN) models to get the embedding of drugs that incorporate structure similarities and use them to create drug pairs' features needed to train the DDI prediction model. GNN is a structured and robust framework for representation learning of graphs. These are neural models that use message passing between graph nodes to extract the dependency of nodes in graph. GNN variants such as graph convolutional network (GCN) [25] and graph attention network (GAT) [26] have exhibited ground-breaking performance on a variety of deep learning tasks in recent years. We utilize GraphSAGE [20], GCN, and GAT to generate drug node embeddings.

Our primary contributions are summarized as follows.

- **Similarity computation**: Chemical substructure is a significant aspect to measure relation and similarity among drugs [23]. If drug pairs have similar chemical structures, drugs are considered to be similar.
- **Appropriate information extraction**: We define the relation between graphs based on important chemical substructures while eliminating less relevant parts of chemical structures. We utilize the ESPF algorithm to extract relevant, influential substructures and create graphs connecting drugs based on frequent, significant chemical substructures.
- **Representation learning**: To construct drug node embeddings that incorporate structure similarity between drugs, we use three different graph neuural network; GCN, GAT, and GraphSAGE. For each drug pair, we concatenate the embeddings from its corresponding drugs and use them as features in ML models.
- **Utilizing proper accuracy measures**: We perform extensive experiments utilizing appropriate accuracy measures, including precision, recall, and F1-score.

The structure of this paper is outlined as follows. First, we present related works for DDI problem in Section II. Then, Section III describes how we create the graph and generate drug embeddings from the graph. Next, we describe our experiments in Section IV, and finally, we give a conclusion in Section V.

II. Related Work

Recently, various computational models have been developed to predict DDIs. Mainly, we categorize them into three groups: similarity-based, classification-based, and graph neural network (GNN)-based methods.

A. *Similarity-based approaches:*

Similarity-based techniques are helpful in predicting DDIs in previous research. These approaches are based on the notion that similar drugs will interact with the same drug [11]. For predicting DDIs, several research studies have used different numbers and types of similarity measures [7], [11], [27], [28]. One noteworthy study is [21], which uses a variety of data sources to generate several local and global similarity measures among drugs. They also admit that their dataset is biased and imbalanced, and they design many experiments to solve these flaws. However, the majority of these approaches take into account limited datasets and fewer drug-centric interactions.

B. *Classification-based approaches:*

In classification-based approaches, different ML models are employed to classify and predict whether two drugs interact or not. For supervised models, drug interactions with other biological entities are used as features. Moreover, some researchers exploit network-based features. For instance, in [29], drug-protein interactions and drug-side effect interactions are used to create a graph. Then they use the created graph to generate a variety of similarity and centrality metrics used as features in different ML models. Likewise, [14] computes feature set based on meta-paths depicting drugs' relation and connecting with other biomedical entities. Afterward, various ML algorithms are applied, and NN is superior to others according to experimental results. [30] uses one-class SVM and KNN to extract valid negative samples. Positive and negative samples are then used for classification and predicting new DDIs using labeled positive and reliable negative samples. Then, they compared their results to those of other baseline approaches and found that their accuracy measures outperformed other baselines. However, many classification-based methods have a few drawbacks, such as not working for new drugs. In our proposed method, we take the significant chemical substructures of drugs into account. A drug pair sharing a vital chemical substructure will likely produce a DDI, regardless of whether or not a drug in the pair is new.

C. *NN-based Approaches:*

Recently, lots of research has predicted drug-related interactions using NNs, especially graph neural networks. PCA representations of drug-mono side effects and drug-protein interactions were combined by [31]. For each drug pair, the corresponding drug features are summed and fed to a NN. For DDI prediction, [32] defines a neural link prediction model as a feed-forward NN that combines node pairs' embeddings to represent a link.

However, drugs' effects are not confined to the molecules they interact with directly in the body. Instead, their impacts are disseminated across the biological networks in which they operate. Thus, GNNs are particularly well adapted to the study of biological networks. [33] constructed a knowledge graph based on protein-protein interaction, drug-drug interaction, and drug-protein interaction. Afterward, they develop a graph convolutional network consisting of encoding, decoding, and model training phases for DDI prediction. [34] utilized GNN to learn drugs and their neighborhood embedding from DDI and the knowledge graph, consisting of interaction among drugs, genes, and proteins. Finally, they predict potential DDIs using binary classification. [35] predicts DDI leveraging the molecular structure of drugs and type of side effects. Drugs are represented as nodes comprising of atoms, with bonds among them represented as edges. Internal messages are sent to each other by nodes/atoms. Furthermore, atoms of different drugs can communicate with one another via outer messages. They calculate an attentional co-efficient for each atom pair, where each atom belongs to a different drug.

III. Methodology

The DDI prediction task entails creating a binary class classification model that takes the feature of two drugs as inputs and generates an output prediction indicating whether or not they interact. While many different types of features are available for drugs, the Simplified Molecular Input Line-Entry System (SMILES) strings [36] are the most common and widely available. Therefore, in our model, we use SMILES strings to create a drugs network and generate more comprehensive features of drugs using GNN. In this section, we present our methodology for DDI predictions. System architecture of this paper is illustrated in Figure 1. Our proposed model consists of four steps:

- Constructing the graph using SMILES strings,
- Drug representation learning from the graph,
- Creating DDI representation,
- Learning with DDI features.

A. Graph construction from SMILES strings

The SMILES string of a drug is a chemical string notation that allows the unique identification of the drug's chemical structure. Among the entire drug's molecular structure, only a few functional sub-structures may cause the chemical reactions resulting in an interaction between drugs, while the remaining substructures are less relevant and/or important for reactions [37]. Hence, in this project we only use frequent sub-structures of drugs. We consider that two drugs are similar and may interact if they have similar functional sub-structures (i.e., functional groups) [23]. By utilizing ESPF algorithm, we extract influential frequent substructures from drugs. Then, we create our graph based on extracted substructures. Basically, we build an edge between two drugs if they share at least a predefined number of sub-structures among them, and thus we construct a drug-drug network. The analysis of the frequent sub-structures allows us to identify the most important or re-occurring parts of each drug, enabling us to build our representations on these most crucial bits of information.

1) Chemical Sequential Pattern Mining: To obtain the frequent sub-structures of the SMILES strings, we employ the ESPF algorithm. ESPF can decompose drugs into a set of customized sub-structures. Given the SMILES strings of drugs as an input, ESPF discovers the frequent recurring sub-structure set and their frequency by repeatedly pairing reoccurring sequential tokens.

We give the ESPF algorithm in Algorithm 1. It starts with a list of tokens as amino acids and atoms in SMILES strings. Then, with considering every pair of the element in this list, it extracts the frequency of these pairs in SMILES strings, and if their frequency is larger than a given threshold, they are converted to a frequent substructure and added to the list. This continues until there is no pair with a frequency higher than the threshold or size of the list exceeds the maximum size.

Algorithm 1: Explainable Substructure Partition Fingerprint (ESPF)

```
Input: A set of initial SMILES tokens $S$ as atoms and bonds, a set of tokenized SMILES strings $T$, a frequency threshold $\theta$, and a size threshold $l$ for $S$.
for $t = 1 \ldots, l$ do
    $(A, B)$, $FREQ \leftarrow$ scan $T$ ; /* $(A, B)$ is the frequentest consecutive tokens. */
    if $FREQ < \theta$ then
        break ; /* $(A, B)$'s frequency lower than threshold */
    end
    $T \leftarrow$ find $(A, B) \in T$, replace with $(AB)$ ; /* update $T$ with the combined token $(AB)$ */
    $S \leftarrow S \cup (AB)$ ; /* add $(AB)$ to the token vocabulary set $S$ */
end
Output: $T$, the updated tokenized proteins/drugs; $S$, the updated token vocabulary set.
```

After extracting frequent sub-structures from SMILE strings, we use them to decompose drugs into a sequence of frequent sub-structures. For a given string, hypothetically, there are several ways to break this string into pieces depending on the vocabulary of sub-structures.

For example, for a given string "Unexplainable" and the given sub-structure set is

{un, plain, une, able, explain, lain, unex, ex, x, ab, bl, le.}

there are several ways to break the string down, such as

Un-ex-plain-able
Un-ex-plain-ab-le
Une-x-plain-able
Une-x-plain-ab-le
Un-explain-able.

Given this variety of possible solutions, we approach the tokenization of the SMILES string set problem by first sorting our vocabulary of sub-structures into a list from most to least frequently appearing. For each SMILES string, we partition in order of frequency with starting from the highest frequency. In the possible case that any small pieces of the SMILES strings do not become part of a pairing, we leave these as-is instead of removing them from the strings. Example of a partitioned SMILES string of a drug;

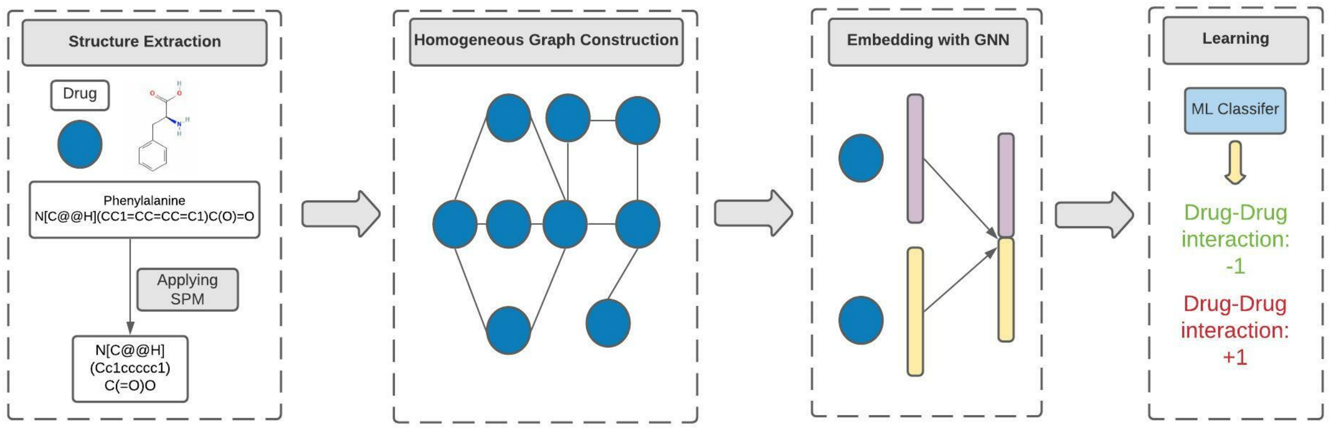
Structure Extraction
Drug
Phenylalanine
N[C@@H](CC1=CC=CC=C1)C(O)=O
Applying SPM
N[C@@H] (Cc1ccccc1) C(=O)O
Homogeneous Graph Construction
Embedding with GNN
Learning
ML Classifer
Drug-Drug interaction: -1
Drug-Drug interaction: +1

Fig. 1. System Architecture of our method

```
DB00224: CC(C)(C) NC(=O)C@@H1
CN(C c2ccc n c2) CCN 1C C@@H(O)C
C@@H( Cc1ccccc1) C(=O)N C@H1
c2ccccc2 CC@H1O
```

2) Graph Creation: After getting the partitioned SMILES strings, we create a weighted graph whose nodes are drugs and add edges between drug nodes if they share a certain number *S* of sub-structures, which we call the graph's S-value. Depending on the S-value, the number of edges changes in the graph. The weight for each edge of the graph is the number of sub-structures that the two-drug nodes shared.

B. Drug Representation learning via Graph

To obtain our representations for the DDI's, we first apply GNN to the graph to learn the node embeddings (each node representing a drug). GNN is a multilayer NN that works with graph structures directly and can generate embeddings of nodes by utilizing global and local structures from graphs [38]. In GNN, information propagates across the edges in the network [39]. GNN can be viewed as a message-passing algorithm where node representations are iteratively computed through passing, transforming, and aggregating the features of their neighbor nodes using a differentiable aggregation function [40], [20].

A layer of a GNN can be expressed as

$$H^{i+1} = f(AH^iW^i)$$

where A is an adjacency matrix of a graph G, $H^{(i)} \in R^{n\times m}$ is a matrix containing node embeddings computed at layer i, $W_i \in R^{m\times k}$ is the trainable weight matrix at layer i that is shared with all vertices in the network, and f is a propagation function that takes adjacency matrix, trainable weight and previous layer node presentations H^{i-1} to produce current layer's node representations.

There are several ways to implement the propagation function f(), among which GCN is one of the most popular variants of GNN where f() is basically a combination of linear feature transformations, aggregations, and point-wise nonlinearities [25]. Each node in GCN is represented by averaging the features of its neighbors as well as its own. Kipf et al. [25] use a spectral method to create propagation rules that sum together information from the previous layer to produce the features of the following layer. Expression of a two-layer GCN is

$$H^1 = Relu(ZH^0W^0),$$

$$H^2 = softmax(ZH^1W^1)$$

where normalized symmetric adjacency matrix, $Z = \hat{D}^{-1/2}\hat{A}\hat{D}^{-1/2}$, and $\hat{A}$ stands for adjacency matrix with 1s on its diagonal for the self-loops. $\hat{D}$ is the degree matrix of adjacency matrix $\hat{A}$, where $\hat{D}_{\text{ii}} = \sum_j \hat{A}_{\text{ij}}$. $H^0 = X$, where X is the initial features of nodes and W^0, W^1 are the trainable weights.

However, GCN is inherently transductive; it requires all the nodes to be present during training, thus failing to work for previously unseen nodes. Moreover, as it needs the whole graph during training, for large networks, it might face scalability issues. To address this issue, GraphSAGE [20] comes with a sampling module. The fundamental idea behind the sampling module is that instead of using all of the information in the neighborhood, we may sample a subset of it for propagation. In particular, it aggregates information by evenly sampling a fixed-size set of neighbors rather than utilizing the whole neighbor set. Moreover, GraphSAGE can be generalized to unseen nodes by inductively learning the embedding for each node. As a result, even if a new node comes into the graph that was unseen during training, its neighbors may still accurately represent it. Mathematically GraphSAGE can be expressed as:

$$h^i_{N_v} = AGG_i(h^{i-1}_u, \forall u \in N_v),$$

$$h^i_v = RELU(W^i \circ [h^i_{N_v} || h^{i-1}_v])$$

where the aggregation function is denoted by AGG. Aggregation can be performed by applying to mean, pooling, and LSTM aggregator [20]. N_v is the neighborhood of node v, and RELU is a non-linear activation function.

In both GCN and GraphSAGE, the neighbor nodes are aggregated to the central node with identical or predefined weights that means all the nodes in the neighborhood get similar importance. However, Graph Attention Network (GAT) [26] assumes that the contribution/importance of all the neighborhood nodes to represent a central node should not be the same; instead, they should be learned during training. Thus GAT modifies the GCN by injecting self-attention mechanism into the propagation stage for each node as follows:

$$h_v^i = \sigma(\sum_{u \in N_v} \alpha_{vu}^i W^i h_u^{i-1})$$

where

$$\alpha_{vu}^i = \frac{\exp(leakyReLU(a^T[W^i h_v^{i-1} || W^i h_u i - 1])}{\sum_{k \in N_v} \exp(leakyReLU(a^T[W^i h_v^{i-1} || W^i h_k^{i-1}])}$$

where α_{vu}^i is node u's contribution to node v's representation in the layer ith, W is the weight matrix associated with each node's linear transformation, a is the weight vector of a single layer feedforward NN, and N_v is the neighborhood of node v. LeakyReLU is a nonlinear activation function. We individually utilize GCN, GAT, and GraphSAGE to generate drug nodes' embeddings of a fixed vector size.

Loss functions: To train and thus minimize the loss between ground truth label and predicted label, a graph-based unsupervised loss function is designed where the loss function promotes adjacent nodes to have comparable embeddings, whereas nodes that are far away are separated in the projected space. Using this method, the nodes will learn more and more about their neighborhood. The graph-based loss function is defined as follows:

$$J_G(h_u) = -\log \sigma(h_u^T h_v) - Q \circ E_{h_n \sim P_n(v)} \log \sigma(h_u^T h_{v_n})$$

where u and v co-occur in a fixed-length random walk, and v_n are the negative samples that do not co-occur with u, σ denotes an activation function, P_n denotes a negative sampling distribution, and Q denotes the number of negative samples.

C. DDI Representation Creation

Once we obtain our drug embeddings from the GNNs, we create our DDI representations. However, though we have a certain amount of positive samples (i.e., sample with known interactions) say 'A', we do not have negative samples (i.e., samples with no interactions) that, along with positive samples, are necessary to train the GNN model. To create the necessary negative samples, we employ random selection from all possible drug pairs without any known interaction. Next, to create our DDI representations, we take the embeddings from its respective involved drugs learned from the GNN and concatenate them.

D. Learning with DDI features

After extracting drug pairs' embeddings using GNN, our objective is to predict whether two drugs interact or not. Now comes the challenge of how to apply drug embeddings and labeled data to the DDI prediction task. DDI prediction can be accomplished by utilizing ML models that can assist computer programs and models in actively learning, evolving, and improving with experience. Any ML algorithm can be utilized to detect the DDI interaction between drug pairs. The ML models we experiment with are Logistic Regression (LR), K-Nearest Neighbors (KNN), and feed-forward neural network (NN). We choose the optimal model by selecting the model that produces the highest accuracy results on the test data.

IV. Experimental Results

A. Dataset

We create our dataset using DrugBank[1]. We select drugs that met certain criteria, namely, drugs with the available following information:

- A SMILES string,
- Known interactions with at least one other drug,
- Inclusion in the group of FDA-approved drugs.

The number of drugs meeting these criteria is 824, and from among these drugs, we have a total of 96,751 known positive drug drug interactions. After collecting the drug set, we canonize each of the SMILES strings to ensure that we work with the recognized standard for SMILES strings. Unfortunately, the SMILES strings that contain the element 'Platinum' specifically are not recognized as "valid" codes. To rectify this issue, we obtain these drugs' canonized SMILES strings from the PubChem database[2] and replace these invalid ones. We use RDKit[3] python package for this process.

We apply the ESPF algorithm to the SMILES strings of our 824 drugs with different frequency thresholds. We observed that when we use a lower threshold, we get many substructures, some of which may not be important, but when we use a higher value, we get fewer sub-structures but may lose some important ones. Based on our experiments, we select five as the frequency threshold to continue with, which generates 741 unique sub-structures from our dataset.

For our Graph, we construct an edge between drugs if they share a certain number (S-value) of sub-structures. We note that if we use 1, we may connect many drugs that may not be related and/or interact with each other. So we start with an S-value of 2 and consider 2, 3 and 4. With the change of S-value, the number of edges in the network also decreases, as does the number of fully connected nodes, which is shown in Table I. We construct three different graphs for our three different S values, then individually apply the three different GNN models on the graphs to get the vector representations of drugs of size 128 each.

[1]https://go.drugbank.com
[2]https://pubmed.ncbi.nlm.nih.gov
[3]https://www.rdkit.org

TABLE I
THE NUMBER OF EDGES AND CONNECTED NODES BASED ON DIFFERENT S-VALUES.

S-value	Number of edges	Number of connected nodes
2	22282	811
3	6025	721
4	2019	527

As noted above, in our dataset, we have 96,751 known interactions among the 824 drugs. These known interactions are positive samples. To generate negative samples, at first we generate all the possible drug pair combinations that results in a 339,076 possible interactions. Of these, 242,325 are not known to have a positive interaction, and from these interactions, we employ random selection to produce a total of 96,751 negative samples to have an even split of training data.

To get the representations of each sample pair (i.e., DDI), we simply concatenate the embeddings of the respective involved drugs, and at the end, we use a label bit 0 or 1 (i.e., ‘0’ for negative samples and ‘1’ for positive samples). Thus, each sample’s positive or negative DDI is represented by a vector of size 257. These representation vectors are used as the features in ML (i.e., KNN, LR and, NN) models for binary classification.

B. Baseline

As a baseline for our DDI representations, we use the CASTER method (CM) outlined in the CASTER paper [23] to create feature vectors for our dataset of positive and negative DDIs. In this paper, each drug is represented by a vector of the same length as the number of unique substructures created from the dataset, with a 0/1 bit corresponding to the existence of each of the unique structures in the drug. To create the DDI representation, these vectors are then logically “and”ed to create the representation for drug pairs. Formal definition of Caster representation is given as follows [23].

Caster Representation: Let $S = S_1, S_2, \ldots, S_k$ is the set of frequent substructures. Each drug pair (D, D') is represented via k-dimensional vector $t = [t_1, \ldots, t_k]$ where $t_i = 1$ if $S_i \in D$ and $S_i \in D'$ and $t_i = 0$ otherwise.

As our set of substructures consists of a total of 741 unique substructures, these DDI representations are of size 742. We use these vectors as the features of drug pairs to feed into the same ML models that we feed our representations. Notice that we only utilize the CASTER DDI representations and not their ML models.

C. Parameter Settings

For each GNN model (GCN, GAT, and GraphSAGE), we use a two-layer architecture with a ’mean’ aggregator in GraphSAGE. The learning rate, batch size, and the number of epochs are set to 0.7, 20, and 5, respectively. For the binary classification, we used a 30% / 70% test/train split on the data. We choose $k = 5$ for the KNN. The NN we use is a simple feed-forward NN with 100 epochs. Both the KNN and the LR models are generic without optimization of parameters. We use these parameter settings for both our models and also for baseline model.

D. Result Analysis

We perform experiments for our models along with three machine learning algorithms; LR, KNN, NN, to see their effect on the results. The performance of our models and baseline methods are assessed using different accuracy measures: accuracy, precision, recall, and F1-score.

We present the performance of our models in Table II and Table III. First, in Table II, we present the detailed performance results for different GNN models for $S = 2$, which we select at the best performing of the S-values, combined with the three distinct machine learning algorithms: LR, KNN and NN. Our experimental results show that GraphSAGE outperforms other embedding methods (GCN, GAT) for all measure and all machine learning methods. It gives the best performance with NN method. The average Accuracy, Precision, Recall, and F1-score of GraphSAGE with NN are 82.39%, 82.19%, 82.83%, and 82.3%, respectively. The average accuracy results for GraphSAGE with NN are around 82%, whereas the average accuracy of GCN and GAT with different ML classifiers is below 80%. Regarding GraphSAGE, inductive learning is used to generate the embedding for each node where aggregation of a node’s neighborhood is used. GraphSAGE has integrated sampling modules. To propagate information through the network, they uniformly sample a subset of the neighborhood information rather than using the entire neighborhood. As a result, even if a new node hidden during training appears in the graph, its neighbors may still appropriately represent it. It produces representable embedding for unseen nodes by aggregating adjacent nodes. It permits node embedding to be used in domains with dynamic graphs, where the graph topology is constantly changing.

According to the results, the NN yields better performance for all GNN models compared to other machine learning approaches. The reason for this performance could be that they can learn and model non-linear and sophisticated connections. Furthermore, after learning from the input data and their connection, they may infer undiscovered associations on previously unseen data. Furthermore, neural networks can discover hidden data connections without enforcing any established association.

We perform another experiment to analyze the impact of S-value (i.e., graph structure) on our model. We use GraphSAGE as GNN model as it gives the best performance in previous experiment. We present our results in Table III. From this table, we see that the performance of our model decreases with the increase of S-value. We find that an S-value of 2 yields the best results overall, with $S = 3$ following and $S = 4$ providing the worst performance. We hypothesize that the main reason for this observed trend is that with the increase of S, the number of connected nodes in the graph decreases and we lose the relation between drugs that have similar functional structure. For example, as given in Table I, for an S-value of

TABLE II
EXPERIMENTAL RESULTS FOR $S = 2$.

Embedding	ML	Accuracy	Precision	Recall	F1
GAT	LR	0.601	0.575	0.775	0.660
GAT	KNN	0.656	0.643	0.703	0.671
GAT	NN	0.775	0.759	0.807	0.782
GraphSAGE	LR	0.617	0.611	0.644	0.627
GraphSAGE	KNN	0.657	0.648	0.687	0.667
GraphSAGE	**NN**	**0.824**	**0.822**	**0.828**	**0.823**
GCN	LR	0.603	0.580	0.753	0.655
GCN	KNN	0.657	0.644	0.702	0.672
GCN	NN	0.774	0.772	0.792	0.779

TABLE III
EXPERIMENTAL RESULTS OF GRAPHSAGE FOR DIFFERENT S VALUES

S value	ML	Accuracy	Precision	Recall	F1
2	LR	0.617	0.611	0.644	0.627
2	KNN	0.657	0.648	0.687	0.667
2	**NN**	**0.824**	**0.822**	**0.828**	**0.823**
3	LR	0.612	0.603	0.656	0.628
3	KNN	0.647	0.640	0.670	0.654
3	NN	0.735	0.739	0.727	0.733
4	LR	0.606	0.619	0.552	0.584
4	KNN	0.632	0.642	0.597	0.619
4	NN	0.685	0.696	0.659	0.677

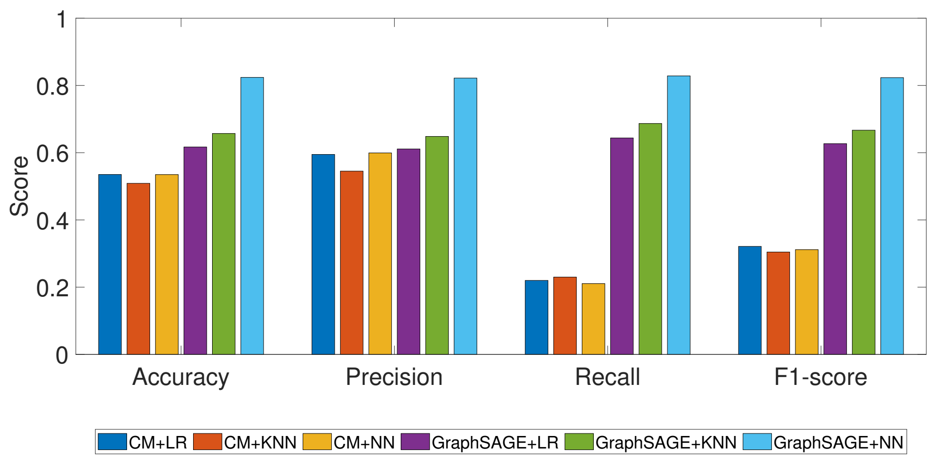

Fig. 2. Performance comparison of the proposed graph based model with the baseline model using LR, KNN and NN machine learning model.

4, only 527 of the total 824 drugs have at least one edge to another node, whereas for S-values of 2 and 3, 811 and 721 drugs, respectively, share at least one edge with other drugs.

Finally, we compare the performance of our model with the baseline model in Figure 2 which uses the CASTER DDI vector representations on the same ML models we run our DDI representations. For our model we use GraphSAGE with $S = 2$ as it gives the best performance. As we see from the figure, our method outperforms the baseline method (CM) significantly with all machine learning models in all accuracy measures. Especially, for recall and F1 measures, the baseline model has a much lower score than our model. Hence, the chemical substructure is critical for DDI prediction, and our usage of frequent chemical substructure information to denote drug relationships in the form of the graph has enabled our model to remain superior to baseline method.

V. Conclusion

In this paper, we propose a novel method to predict drug-drug interactions using only the chemical structure of drugs through their SMILES representations. First, we construct a graph leveraging relations of drugs based on their common frequent chemical substructures we extract from the SMILES string. Utilizing three different GNN models, we generate drug embeddings from the graph and concatenate the drug embeddings to create features of the DDI pairs. Finally, we apply different machine learning algorithms to the created drug pair features to create DDI prediction models. We perform extensive experiments to evaluate our model and compare our results with a baseline model. According to our results, our method gives more accurate results than the baseline model using the CASTER DDI representations with GraphSage as GNN, NN as machine learning and $S = 2$ as the relation threshold.

For future work, we plan to extend our model to predict exact side effect(s) caused by DDIs. We also plan improve negative pair selection by learning from known non-interacting pairs to generate a full set on negative DDIs instead of employing Random Selection, which we are currently doing. Last, but not least, we plan to look into creation of a hypergraph from our substructures and using HyperGraph Nueral Network Learning.

Acknowledgment

This research was supported by Research Experience for Undergraduates (REU) program through the National Science Foundation grant no. 2050978.

References

[1] J. Jia, F. Zhu, X. Ma, Z. Cao, Y. Li, and Y. Z. Chen, "Mechanisms of drug combinations: interaction and network perspectives," *Nature Reviews Drug Discovery*, vol. 8, pp. 111–128, 2009.

[2] K. Han, E. E. Jeng, G. T. Hess, D. Morgens, A. Li, and M. C. Bassik, "Synergistic drug combinations for cancer identified in a crispr screen for pairwise genetic interactions," *Nature biotechnology*, vol. 35, pp. 463 – 474, 2017.

[3] M. Bansal, J. Yang, C. Karan, M. P. Menden, J. C. Costello, H. Tang, G. Xiao, Y. Li, J. D. Allen, R. Zhong, B. Chen, M. Kim, T. Wang, L. M. Heiser, R. B. Realubit, M. Mattioli, M. J. Alvarez, Y. Shen, D. Gallahan, D. Singer, J. Saez-Rodriguez, Y. Xie, G. Stolovitzky, and A. Califano, "A community computational challenge to predict the activity of pairs of compounds," *Nature Biotechnology*, vol. 32, pp. 1213–1222, 2014.

[4] J. Strandell, A. Bate, M. Lindquist, and I. R. Edwards, "Drug-drug interactions - a preventable patient safety issue?" *British journal of clinical pharmacology*, vol. 65 1, pp. 144–6, 2008.

[5] S. Huang, R. Temple, D. Throckmorton, and L. Lesko, "Drug interaction studies: Study design, data analysis, and implications for dosing and labeling," *Clinical Pharmacology & Therapeutics*, vol. 81, 2007.

[6] Y. Zheng, H. Peng, X. Zhang, Z. Zhao, J. Yin, and J. Li, "Predicting adverse drug reactions of combined medication from heterogeneous pharmacologic databases," *BMC Bioinformatics*, vol. 19, 2018.

[7] S. Vilar, E. Uriarte, L. Santana, N. P. Tatonetti, and C. Friedman, "Detection of drug-drug interactions by modeling interaction profile fingerprints," *PLoS ONE*, vol. 8, 2013.

[8] E. D. Kantor, C. D. Rehm, J. S. Haas, A. T. Chan, and E. L. Giovannucci, "Trends in prescription drug use among adults in the united states from 1999-2012." *JAMA*, vol. 314 17, pp. 1818–31, 2015.

[9] F. R. Ernst and A. J. Grizzle, "Drug-related morbidity and mortality: updating the cost-of-illness model." *Journal of the American Pharmaceutical Association*, vol. 41 2, pp. 192–9, 2001.

[10] X. Lin, Z. Quan, Z.-J. Wang, H. Huang, and X. Zeng, "A novel molecular representation with bigru neural networks for learning atom," *Briefings in bioinformatics*, 2020.

[11] S. Vilar, R. Harpaz, E. Uriarte, L. Santana, R. Rabadán, and C. Friedman, "Drug-drug interaction through molecular structure similarity analysis," *Journal of the American Medical Informatics Association : JAMIA*, vol. 19 6, pp. 1066–74, 2012.

[12] T. Ma, C. Xiao, J. Zhou, and F. Wang, "Drug similarity integration through attentive multi-view graph auto-encoders," in *IJCAI*, 2018.

[13] J. Y. Ryu, H. U. Kim, and S. Y. Lee, "Deep learning improves prediction of drug–drug and drug–food interactions," *Proceedings of the National Academy of Sciences*, vol. 115, pp. E4304 – E4311, 2018.

[14] F. Tanvir, M. I. K. Islam, and E. Akbas, "Predicting drug-drug interactions using meta-path based similarities," in *CIBCB*, 2021.

[15] B. Jin, H. Yang, C. Xiao, P. Zhang, X. Wei, and F. Wang, "Multitask dyadic prediction and its application in prediction of adverse drug-drug interaction," in *AAAI*, 2017.

[16] X. Chu, Y. Lin, Y. Wang, L. Wang, J. Wang, and J. Gao, "Mlrda: A multi-task semi-supervised learning framework for drug-drug interaction prediction," in *IJCAI*, 2019.

[17] Z. Quan, X. Lin, Z.-J. Wang, Y. Liu, F. Wang, and K. Li, "A system for learning atoms based on long short-term memory recurrent neural networks," *2018 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, pp. 728–733, 2018.

[18] Y. Le, Z.-J. Wang, Z. Quan, J. He, and B. Yao, "Acv-tree: A new method for sentence similarity modeling," in *IJCAI*, 2018.

[19] Z. Quan, Z.-J. Wang, Y. Le, B. Yao, K. Li, and J. Yin, "An efficient framework for sentence similarity modeling," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 27, pp. 853–865, 2019.

[20] W. L. Hamilton, R. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, 2017, pp. 1025–1035.

[21] I. Abdelaziz, A. Fokoue, O. Hassanzadeh, P. Zhang, and M. Sadoghi, "Large-scale structural and textual similarity-based mining of knowledge graph to predict drug-drug interactions," *J. Web Semant.*, vol. 44, pp. 104–117, 2017.

[22] S. Jaeger, S. Fulle, and S. Turk, "Mol2vec: Unsupervised machine learning approach with chemical intuition," *Journal of chemical information and modeling*, vol. 58 1, pp. 27–35, 2018.

[23] K. Huang, C. Xiao, T. N. Hoang, L. Glass, and J. Sun, "Caster: Predicting drug interactions with chemical substructure representation," in *AAAI*, 2020.

[24] L. G. J. S. Kexin Huang, Cao Xiao, "Explainable substructure partition fingerprint for protein, drug, and more," *NeurIPS Learning Meaningful Representation of Life Workshop*, 2019.

[25] T. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," *ArXiv*, vol. abs/1609.02907, 2017.

[26] P. Velickovic, G. Cucurull, A. Casanova, A. Romero, P. Lio', and Y. Bengio, "Graph attention networks," *ArXiv*, vol. abs/1710.10903, 2018.

[27] A. Gottlieb, G. Y. Stein, Y. Oron, E. Ruppin, and R. Sharan, "Indi: a computational framework for inferring drug interactions and their associated recommendations," *Molecular Systems Biology*, vol. 8, pp. 592 – 592, 2012.

[28] P. Zhang, F. Wang, J. Hu, and R. Sorrentino, "Label propagation prediction of drug-drug interactions based on clinical side effects," *Scientific Reports*, vol. 5, 2015.

[29] B. Davazdahemami and D. Delen, "A chronological pharmacovigilance network analytics approach for predicting adverse drug events," *Journal of the American Medical Informatics Association*, vol. 25, p. 1311–1321, 2018.

[30] Y. Zheng, H. Peng, X. Zhang, Z. Zhao, X. Gao, and J. Li, "Ddi-pulearn: a positive-unlabeled learning method for large-scale prediction of drug-drug interactions," *BMC Bioinformatics*, vol. 20, 2019.

[31] R. Masumshah, R. Aghdam, and C. Eslahchi, "A neural network-based method for polypharmacy side effects prediction," *BMC Bioinformatics*, vol. 22, 2021.

[32] G. K. O. Crichton, Y. Guo, S. Pyysalo, and A. Korhonen, "Neural networks for link prediction in realistic biomedical graphs: a multi-dimensional evaluation of graph embedding-based approaches," *BMC Bioinformatics*, vol. 19, 2018.

[33] M. Zitnik, M. Agrawal, and J. Leskovec, "Modeling polypharmacy side effects with graph convolutional networks," *Bioinform.*,

vol. 34, no. 13, pp. i457–i466, 2018. [Online]. Available: https://doi.org/10.1093/bioinformatics/bty294

[34] X. Lin, Z. Quan, Z. Wang, T. Ma, and X. Zeng, “Kgnn: Knowledge graph neural network for drug-drug interaction prediction,” in *IJCAI*, 2020.

[35] A. Deac, Y.-H. Huang, P. Velickovic, P. Lio’, and J. Tang, “Drug-drug adverse effect prediction with graph co-attention,” *ArXiv*, vol. abs/1905.00534, 2019.

[36] System, “System, D. C. I. 2015. Smiles tutorial.”

[37] R. B. Silverman and M. W. Holladay, *The organic chemistry of drug design and drug action*. Academic press, 2014.

[38] Z. Zhang, P. Cui, and W. Zhu, “Deep Learning on Graphs: A Survey,” *IEEE Transactions on Knowledge and Data Engineering*, 2020.

[39] Q. Xie, J. Huang, P. Du, M. Peng, and J.-Y. Nie, “Graph topic neural network for document representation,” in *Proceedings of the Web Conference 2021*, 2021, pp. 3055–3065.

[40] J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals, and G. E. Dahl, “Neural message passing for quantum chemistry,” in *International conference on machine learning*. PMLR, 2017, pp. 1263–1272.