Early Detection of Drug Toxicity using Neural Networks

Ishana Saroha

ABSTRACT

Toxicity prediction of drugs is a critical step in the drug development process, as it evaluates the safety of drugs. An abundance of resources go into the development of new drugs, yet only 12% of all drugs are considered by the FDA, since many potential drugs are toxic. By the time toxicity has been identified in conventionally developed drugs, anywhere from $1 to $2 billion could've been invested. Therefore, it is imperative to have an early detection of drug toxicity. AI can be used to predict toxicity, by using QSAR and machine learning methods. Some ML based (DNN, SVM, RF) solutions have been proposed that use only molecular features of the compounds, not molecular structure. The goal of this project was to create a GNN ML model (which uses both atomic features and molecular structure information) that could make better toxicity predictions. Tox21 toxicity dataset was used for all training and evaluation. Three GNN models were created and then compared to a SVM model. Statistical analysis of results showed that the GNN models performed better than the SVM model, with the GNN models having better F1-scores (3.34%-6.44% improvement) and MCC values (5.48%-9.40% improvement). Results of this project show that GNN-based models have better toxicity prediction compared to SVM-based models. GNNs have great potential to augment existing QSAR methods used in predicting toxicity in the drug development industry, thus reducing cost, time and resources invested, and alleviating ethical concerns of animals/clinical trials when compared to conventionally developed drugs. This model can help in molecule toxicity prediction for pharmaceutical and other industries.

KEYWORDS

Drug toxicity, Artificial Intelligence (AI), Machine learning, QSAR, Graph Neural Networks(GNN), SVM, Tox21

INTRODUCTION

Toxicity prediction of drugs is a critical step in the drug development process, as it evaluates the safety and effectiveness of the drug. Every year, on average, the Food and Drug Administration (FDA) grants approval to 38 new drugs (Congressional Budget Office [CBO], 2021), permitting production of the drug and allowing it to be distributed to the general public. The average cost to develop a new drug can be anywhere from $1 billion to $2 billion (CBO, 2021), as it generally requires 10 to 15 years of testing and clinical trials (MatchTrial, n.d.). An abundance of resources, money, and time goes into the drug development process, yet only 12% of all drugs are considered by the FDA (CBO, 2021). A major reason for this is because many potential drugs are toxic, and can no longer be pursued. By the time toxicity is identified, an abundance of time and resources have been invested into the drug. If toxicity is not detected early in the development process, it could lead to the loss of millions of dollars and years of research that go into drugs' preclinical and clinical trials, using animal models and human testing. Therefore, it is imperative to have early detection of drug toxicity. Artificial Intelligence

(AI) can make this possible, by using Structure Activity Relationships (SARs) and machine learning methods. SARs are being used to predict activity of new molecules, by studying relationships between chemicals with similar molecular structures (OECD, n.d.), and comparing their biological activity. Graph Neural Networks (GNNs) are a class of deep learning methods designed to perform inference on data described by graphs. It is hypothesized that a GNN model can be written and trained to have better performance than an SVM-based classifier. The project will create a neural network in Python. It will be trained using a training dataset and will be evaluated based on the accuracy of its prediction for the testing dataset, with known toxicity information. The control variables will be the training and testing datasets. The independent variables are the network parameters and the dependent variable is the prediction accuracy of the model.

Due to advances in technology and scientific methods, thousands of new chemical compounds have been created in past years, many of which could potentially be harmful to human health (Columbia Mailman School, 2020). Toxicology is a scientific discipline, with connections to the fields of biology, chemistry, pharmacology, and medicine. It is the study of how chemicals, biological agents, and physical agents impact living organisms, specifically humans (Columbia Mailman School, 2020). The purpose of toxicology studies is to understand the impact these chemical compounds have on the environment, human body, and beyond. They also help to evaluate the safety of potential drugs, with the usage of animal models and validated procedures (Dorato & Buckley, 2007). Animal reactions are translated, in order to understand risk for humans. In addition to this, toxicologists/researchers measure and analyze substances, particles, pollutants, and bacteria, in order to identify substances that pose possible threats to humans (Columbia Mailman School, 2020). They also detect chemicals that can or cannot be used in medicine, construction, and air and water control. All of this information can help toxicologists determine proper dosage amounts and safe exposure limits (Columbia Mailman School, 2020). Columbia Mailman School explains that by using this information, toxicologists can “uncover the adverse effects of medical treatments and establish … exposure guidelines for substances" (2020).

A major component of toxicology is to study drugs, from their safety to efficiency to effectiveness. Drugs are chemicals used for the purpose of diagnosis, treatment, or prevention of a disease (Pelikan, 2022). They modify the behavior of cells exposed to the drug, by increasing or decreasing the magnitude or frequency of the duration of the normal functions of the cell (Pelikan, 2022). The process of developing drugs is extremely lengthy and costly, often requiring an abundance of time, money, and other resources. During the discovery/development phase, researchers work to devise new drugs through new insights into a disease process, which requires multiple tests of molecular compounds and looking into existing treatments/new technologies, followed by experiments to learn more about the benefits, side effects, and other properties of the drug (FDA, n.d.). After this, researchers move to step two, preclinical research. During this time, they are testing/measuring toxicity of the drug. This research must provide detailed info on dose and toxicity levels. Step three is clinical research, during which the drugs are tested on people to ascertain safety and effectiveness (FDA, n.d.). These studies are designed to test specific research questions regarding the drug. After each phase of clinical research, a smaller percentage of drugs move on to the next. If a drug passes step three, it is brought to the Food and Drug Administration (FDA), for FDA approval (FDA, n.d.). Overall, only about 12% of all drugs are considered by the FDA (Verboon, 2021), showing how strict the guidelines are for carrying out additional research. Overall, the entire process of developing a

safe drug can take anywhere from 10 to 15 years (MatchTrial, 2020), and on average costs the company $1.7 billion (Guengerich, 2010). If toxicity is not detected early on in the drug development process, it could lead to the loss of millions of dollars and years of research that go into drugs' preclinical and clinical trials, using animal models and human testing.

During the process of drug development, the main objective of toxicology studies is to determine if potential drug candidates are safe. Toxicity and safety testing is performed on the potential drugs multiple times through the development process (Guengerich, 2010). There are multiple contexts of toxicity that are observed during these studies, including On-Target Toxicity, Hypersensitivity and Immune Responses, Off-Target Toxicity, Bioactivation and Idiosyncratic Reactions. On-Target Toxicity is due to the interaction of the drug with the same target that produces the desired pharmacological response (Guengerich, 2010). The concept is that the "biological response that the drug exhibits upon binding to its target is the same one that produces both the efficacious and the toxic effects" (Guengerich, 2010). Hypersensitivity and Immune Responses are another context of toxicity, during which the drugs/metabolites (reactive products of metabolism) react with proteins in the body as haptens, to develop antibodies and immune responses (Guengerich, 2010). During Off-Target Toxicity, the drug is not specific with its interactions, causing it to bind to an alternate target and cause toxicity. The drug is nonspecific due to its complexity (Guengerich, 2010). During Bioactivation, the drug is converted into metabolites. This modifies the proteins it reacts with, and somehow causes toxicity (Guengerich, 2010). One possibility for this occurring is that modified regulatory/other proteins lose function. Another possibility is that modified proteins cause immune responses, which is connected to the second context of toxicity, Hypersensitivity and Immune Responses (Guengerich, 2010). The last context of toxicity is Idiosyncratic Reactions. Idiosyncratic means individual, so this context is not the same for every person. It is also very rare, occurring in every 1/103 to 1/104 patients (Guengerich, 2010). This is extremely problematic as animal models are not very predictive, so it is hard to find these adverse events in clinical trials. However, with widely-used drugs (millions of prescriptions), 1/104 occurrence is severe and yields hundreds of problems (Guengerich, 2010). All of these types of toxicity can result in severe problems if they go undetected, making it extremely important to be researched. However, many times, these issues are not detected until late in the development process, making it especially crucial to develop an accurate machine learning model to detect this toxicity early on in the development process.

In order to create a machine learning model that is capable of detecting toxicity in potential drugs, an approach known as Structure Activity Relationships, or SARs, will be utilized. SARs are designed to determine relationships between the chemical structure/structure-related properties and biological activity/target property of compounds (OECD, n.d.). They link chemical structure to chemical properties, such as water solubility, or biological activity, such as toxicity (OECD, n.d.). Qualitative/Quantitative Structure Activity Relationships, or QSARs, are mathematical relationships that relate SAR properties to the "presence or absence, or potency of another property or activity of interest" (OECD, n.d.). Qualitative structure activity relationships are derived from non-continuous data, while quantitative structure activity relationships are derived from continuous data (OECD, n.d.). The main assumption here is that the activity or property of a molecule is reflected and can be predicted through its structure. Similar molecules have similar properties, assuming that the structure dictates the features responsible for its physical, chemical, and biological properties (OECD, n.d.). They depend on the ability to represent the chemical by at least one descriptor. The development of QSARs

requires a data set that provides activity (typically measured through an experiment) for a group of chemicals, structural criteria or structure-related property data set for the same group of chemicals, and a way to relate the two groups of data arrays (OECD, n.d.). The first step of this process is to identify whether or not there is an SAR between molecules in the data collection, based on their associated activities (Guha, 2013). This information is then used to make "structural modifications to optimize some property or activity" (Guha, 2013). This understanding helps researchers "rationally explore chemical space, which … is essentially infinite" (Guha, 2013), allowing them to make predictions for a variety of chemicals and molecules. The only issue with using SAR/QSAR models is that it can be difficult to accurately model differences at molecular level, since any molecular difference can create changes in activity of the chemical (OECD, n.d.).

AI and Machine learning has been used in QSAR analysis for a while. Currently used algorithms include k-nearest neighbors, Naive bayes, Random forest, Support Vector Machine (SVM) and Neural network (Lo et al., 2018.). AI is the ability of a machine to perform cognitive tasks commonly associated with human minds (Britannica, 2023). It is often used for developing systems that contain similar intelligent characteristics of humans (Britannica, 2023). Generalization is a type of AI learning, during which past experiences are used to analyze new experiences. Machine learning is a subfield of AI that employs generalization processes. It gives the computer the ability to learn and solve new problems, without being explicitly programmed (Brown, 2021). To do this, labeled training data is collected, which is used to train the machine learning program, without the intervention of human programmers (Brown, 2021). The more data in this dataset, the more accurate the program will be, as it lowers the chance of overfitting the data. Next, programmers choose a machine learning program to use the input the data, and let the model train itself to identify patterns in the data or make predictions (Brown, 2021). The programmer can adjust the parameters to help make results more accurate. After this, a second dataset, known as the testing or evaluation set, tests how accurate the model is when given new data. Now, the model is ready to be used with a new set of data in the future. This is known as supervised machine learning, as the model is trained with labeled data sets (Brown, 2021).

Support Vector Machines (SVMs) are used to maximize the accuracy of predictions of models by "mapping data to a high-dimensional feature space so that data points can be categorized, even when the data are not otherwise linearly separable"(IBM, 2021). Data is transformed using Kernel functions so that the separator can "be drawn as a hyperplane" (IBM, 2021). Popular Kernel functions include Linear, Polynomial, Radial Basis Function (RBF) and Sigmoid. SVMs are especially useful for data with large numbers of predictor fields such as in text mining, bioinformatics and protein structure prediction (IBM, 2021).

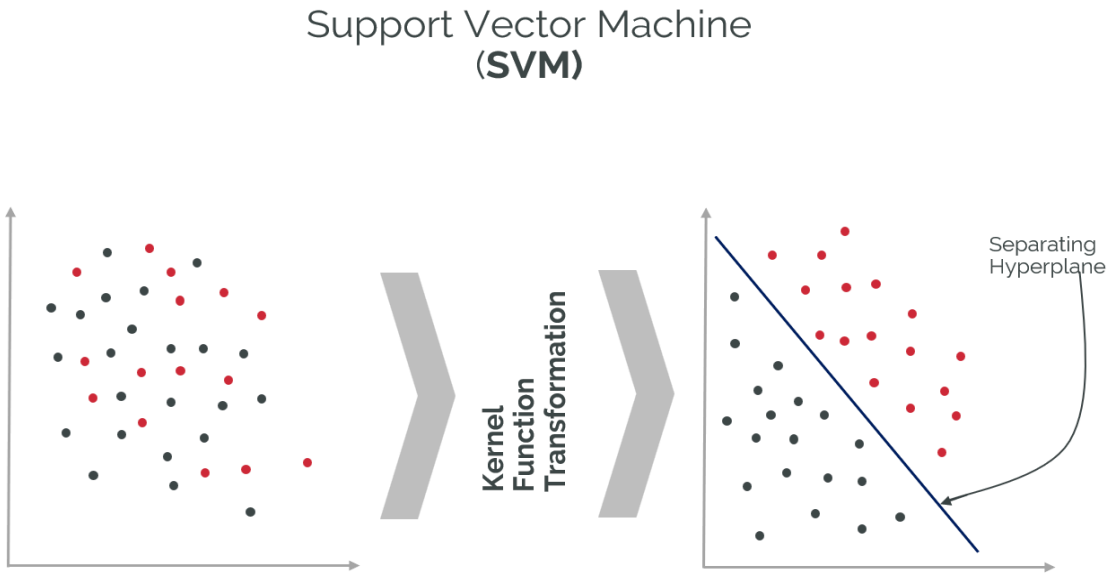

Figure 1. Example of Support Vector Machine (SVM). By Ishana Saroha.

Deep learning is a subset of machine learning, in which algorithms and large sets of data are used to find patterns and create outputs (Western Governors University [WGU], 2020). This requires the use of neural networks. Neural Networks, or Artificial Neural Networks (ANNs), are connections made to imitate the human brain, by taking in information and then generating an output based on its knowledge (WGU, 2020). This helps the model adapt and learn, without the use of needing to be coded repeatedly (WGU, 2020).

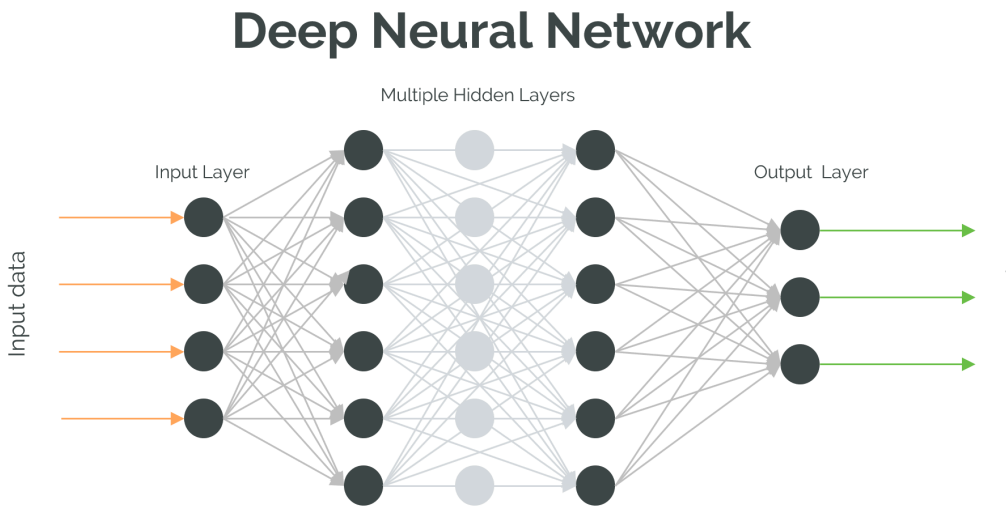

Figure 2. Example of Deep Neural Network. By Ishana Saroha.

ANNs have multiple node layers: an input layer, at least one hidden layer, and an output layer (IBM, n.d. c). Each individual node, or artificial neuron, has its own linear regression model, and all of them are connected. Every node has associated input data, weight, threshold and output (IBM, n.d. c). If the input of any node is above the threshold for that node, then the node is activated, sending data to the next layer of the network. Otherwise, no information is sent (IBM, n.d. c). Each node inside the neural network is responsible for solving a part of the overall problem; each one passes on its information to the next, until the model has developed an output (WGU, 2020). These neurons typically use trial and error to solve the problem (WGU, 2020), and they depend on training data to learn and improve their accuracy over time (IBM, n.d. c). In essence, neural networks learn to map given inputs to a predicted output, using training data. This can be expressed as a function

$$Z = F(X)$$, where Z is the output classification of input data X.

Deep learning can have hundreds or thousands of these layers, to help handle large amounts of data (WGU, 2020). This is just the overall structure of neural networks; there are many different types of neural networks used for different problem domains, e.g. RNN (Recursive Neural Network) used for language translation/natural language processing, speech recognition (IBM, n.d. b), and CNN (Convolution Neural Network) used for image classification and object recognition (IBM, n.d. a).

Deep neural networks have already been used successfully in drug toxicology prediction. In 2014, the Tox21 Data Challenge was won by DeepTox. The Toxicology in the 21st Century (Tox21) program, a federal collaboration involving NIH, the Environmental Protection Agency, and the Food and Drug Administration, was aimed at developing better toxicity assessment methods. The goal of the challenge is to "crowdsource" data analysis by independent researchers to reveal how well they can predict compounds' interference in biochemical pathways using only chemical structure data. (“Tox21 Data Challenge, 2014,” 2014). DeepTox tested different numbers of layers and different numbers of hidden features in those layers. In the end, Deep-learning based DeepTox pipeline outperformed all competitors.

	AVG	NR	SR	AhR	AR	AR-LBD	ARE	Aromatase	ATAD5	ER	ER-LBD	HSE	MMP	p53	PPAR.g
our method	**0.846**	**0.826**	**0.858**	**0.928**	0.807	**0.879**	**0.840**	0.834	0.793	**0.810**	0.814	**0.865**	0.942	0.862	**0.861**
AMAZIZ	0.838	0.816	0.854	0.913	0.770	0.846	0.805	0.819	**0.828**	0.806	0.806	0.842	**0.950**	0.843	0.830
dmlab	0.824	0.811	0.850	0.781	**0.828**	0.819	0.768	**0.838**	0.800	0.766	0.772	0.855	0.946	**0.880**	0.831
T	0.823	0.798	0.842	0.913	0.676	0.848	0.801	0.825	0.814	0.784	0.805	0.811	0.937	0.847	0.822
microsomes	0.810	0.785	0.814	0.901	–	–	0.804	–	0.812	0.785	**0.827**	–	–	0.826	0.717
filipsPL	0.798	0.765	0.817	0.893	0.736	0.743	0.758	0.776	–	0.771	–	0.766	0.928	0.815	–
Charite	0.785	0.750	0.811	0.896	0.688	0.789	0.739	0.781	0.751	0.707	0.798	0.852	0.880	0.834	0.700
RCC	0.772	0.751	0.781	0.872	0.763	0.747	0.761	0.792	0.673	0.781	0.762	0.755	0.920	0.795	0.637
frozenarm	0.771	0.759	0.768	0.865	0.744	0.722	0.700	0.740	0.726	0.745	0.790	0.752	0.859	0.803	0.803
ToxFit	0.763	0.753	0.756	0.862	0.744	0.757	0.697	0.738	0.729	0.729	0.752	0.689	0.862	0.803	0.791
CGL	0.759	0.720	0.791	0.866	0.742	0.566	0.747	0.749	0.737	0.759	0.727	0.775	0.880	0.817	0.738
SuperTox	0.743	0.682	0.768	0.854	–	0.560	0.711	0.742	–	–	–	–	0.862	0.732	–
kibutz	0.741	0.731	0.731	0.865	0.750	0.694	0.708	0.729	0.737	0.757	0.779	0.587	0.838	0.787	0.666
MML	0.734	0.700	0.753	0.871	0.693	0.660	0.701	0.709	0.749	0.750	0.710	0.647	0.854	0.815	0.645
NCI	0.717	0.651	0.791	0.812	0.628	0.592	0.783	0.698	0.714	0.483	0.703	0.858	0.851	0.747	0.736
VIF	0.708	0.702	0.692	0.827	0.797	0.610	0.636	0.671	0.656	0.732	0.735	0.723	0.796	0.648	0.666
Toxic Avg	0.644	0.659	0.607	0.715	0.721	0.611	0.633	0.671	0.593	0.646	0.640	0.465	0.732	0.614	0.682
Swamidass	0.576	0.596	0.593	0.353	0.571	0.748	0.372	0.274	0.391	0.680	0.738	0.711	0.828	0.661	0.585

Figure 3. Tox21 challenge result. From *Table 5, THE LEADING TEAMS' AUC RESULTS ON THE FINAL TEST SET IN THE TOX21 CHALLENGE, DeepTox: Toxicity Prediction using Deep Learning.*

This project will make use of graph neural networks, or GNNs. GNNs are used when data is derived from non-Euclidean domains and are shown using graphs with complex relationships (Menzli, 2023). In computer science, graphs are data structures that have nodes (vertices) and edges (Menzli, 2023). They represent sets of objects and their connections, where relations are edges and entities are nodes (Sanchez-Lengeling et al., 2021). Each one is embedded with data, and can be specialized by adding directionality to edges (directed vs undirected edge) (Sanchez-Lengeling et al., 2021).

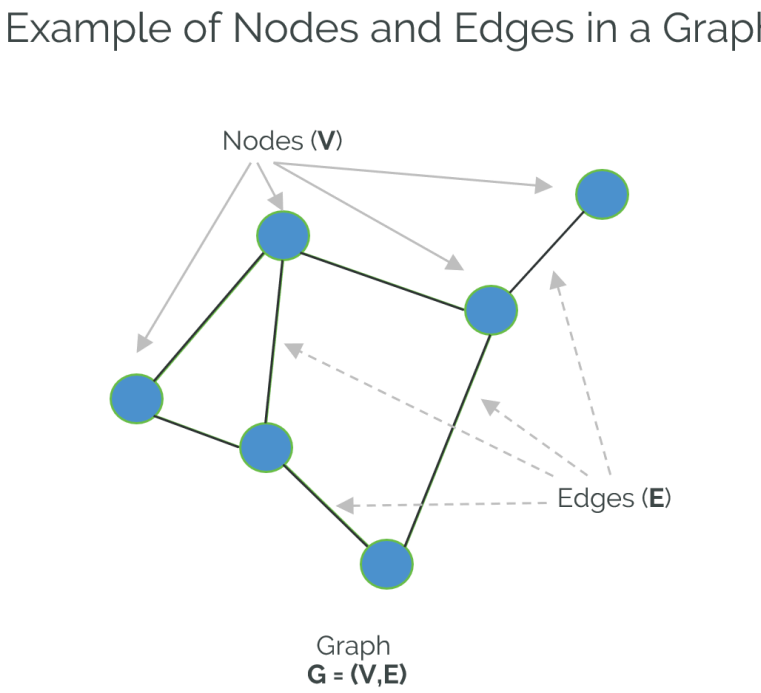

Figure 4. Example of nodes and edges. By Ishana Saroha.

Graph **G** can be represented as (Menzli, 2023)

G = (**V, E**), where **V** = set of nodes and **E** = edges in between nodes.

Graphs can be hard to analyze, as they are complex and have undefined forms and unordered nodes. Therefore, they must be interpreted using GNNs (Menzli, 2023). GNNs are very useful when the amount of neighbors to each node is variable, such as in molecules (Sanchez-Lengeling et al., 2021). Nodes represent atoms and edges represent covalent bonds, and different pairs of atoms and bonds have different varying distances (Sanchez-Lengeling et al., 2021). While CNNs and RNNs are more suited for text, speech and images, GNNs are explicitly made to process graph datasets like social media connections, molecule structures, and citation networks (Labonne, 2023). GNNs can be used for node classification, link prediction and graph classification.

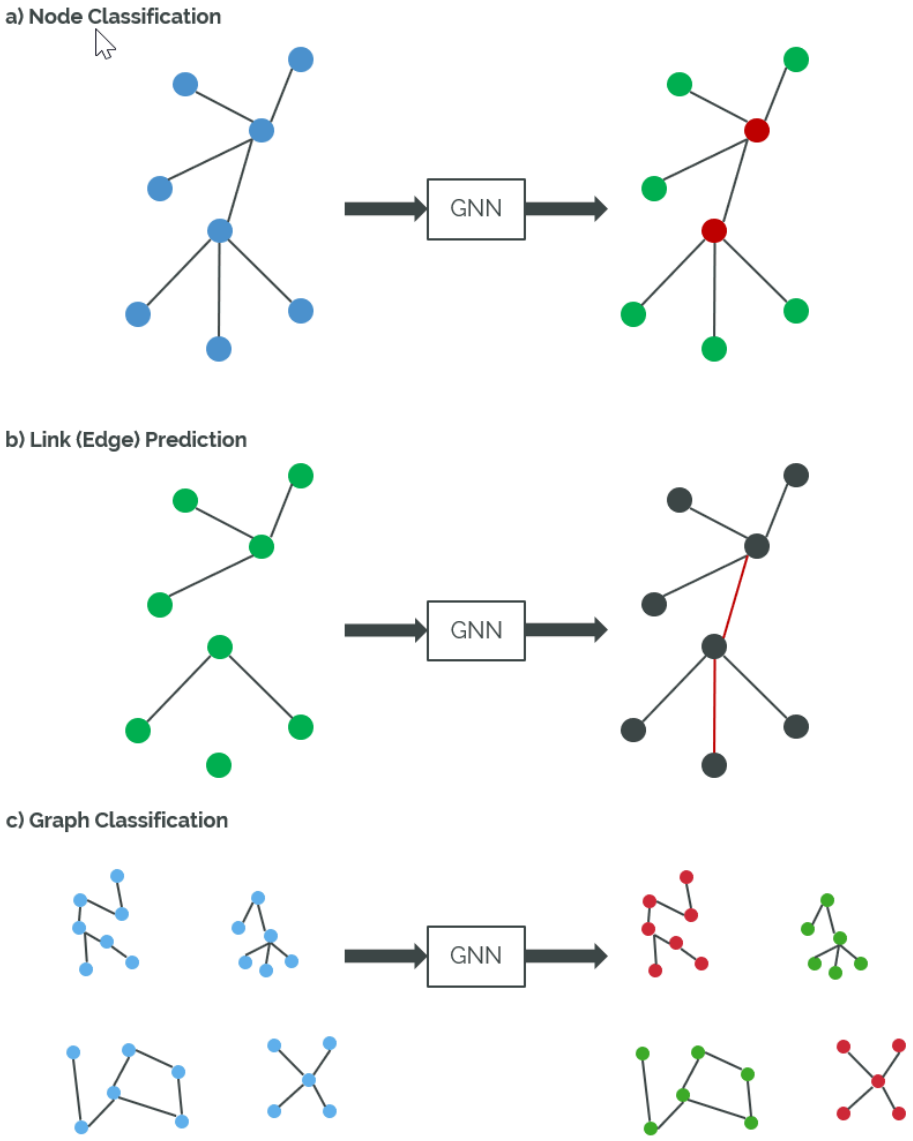

Figure 5. Examples of Graph Datasets and GNNs. By Ishana Saroha.

GNN works by aggregating information from neighboring nodes to create an alternate representation of each node. Thus, this alternate representation of a node contains information not only from itself but also from its neighbors and their neighbors. This way all nodes end up with information from all other nodes. This alternate representation can then be fed into an ANN model to make predictions and classification.

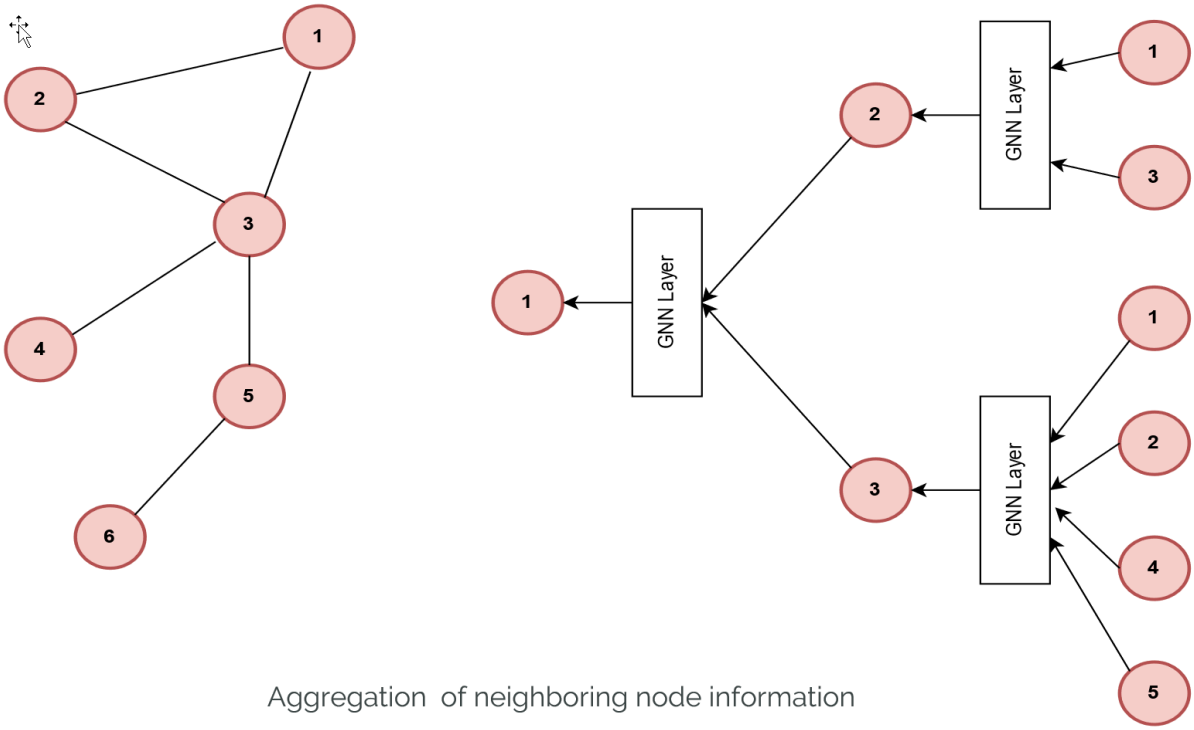

Figure 6. Aggregation of neighboring node information. By Ishana Saroha.

METHODS

To begin preparing data for training and testing models, a new Jupyter Notebook was created in Visual Studio Code and saved as *data_preprocessing.ipynb*. The Tox21 dataset was downloaded using the PyTorch Geometric MoleculeNet function. Molecular fingerprints were then generated for each compound using RDKit, based on their SMILES identifiers; these features were intended for use in training a Support Vector Machine (SVM) model. The dataset was split into training and testing sets in an 80:20 ratio, and both subsets were saved to disk as separate files. This prepared data was then ready for use in both Graph Neural Network (GNN) and SVM models.

To implement the Graph Neural Network (GNN) model, a neural network class was defined with configurable parameters, including the number of hidden features, Graph Convolution layers, Linear layers, and the size of the Linear layer outputs. A training function was added, accepting the GNN model, training dataset, loss function, and optimizer as arguments. This function calculated the loss over the full dataset and returned the mean loss. A testing function was also created to evaluate model performance using a testing dataset, returning the prediction accuracy. A combined function was then implemented to train and test the model over 2000 epochs, with early stopping if the training loss plateaued. Throughout training, loss values, accuracy scores, and total training time were tracked and saved. Once training concluded, the final model was saved to disk. The model's performance was further evaluated by computing ROC curve parameters and the AUC score using the scikit-learn library,

with all results recorded in a data table. Finally, model parameters were modified, and the entire process was repeated to optimize performance.

To implement and train a Support Vector Machine (SVM) model, a classification model was written with configurable parameters to allow experimentation and optimization. The model included control over the regularization parameter, which adjusts the trade-off between achieving a low training error and maintaining a margin that avoids overfitting. Kernel types such as radial basis function (RBF), linear, and polynomial (poly) were incorporated to transform input data into higher-dimensional spaces where separation is more feasible. The gamma parameter, which defines kernel coefficients for RBF and poly kernels, was also adjustable. The model was trained and tested using the molecular fingerprint data generated earlier. A function was written to evaluate the model's performance by recording accuracy scores, calculating ROC curve parameters, and computing the AUC score alongside the SVM parameters used. This function was then called with a variety of parameter combinations, and results for each were recorded. The configuration yielding the highest accuracy and AUC score was selected for comparison against the Graph Neural Network (GNN) model to determine relative performance.

RESULTS

kernel	value of reg_param	gamma	class_weight	train_time	F1 score	MCC value
RBF	10	auto	balanced	0h : 2min : 55.2274sec	0.577903683	0.520336119
linear	50	auto	balanced	0h : 1min : 37.6678sec	0.454258675	0.379696217

Table 1. SVM models and Parameters. This table shows parameters of the two best SVM models. The best SVM model (highlighted) was used for the final experiment.

name_gnn_class	Num GCN layers	GCN Layer Type	num_hidden_features	num_linear_layers	num_hidden_lin_features	num_parameters	optimizer	learning_rate	Batch size	F1 score	MCC value
GCN_2_dropout	3	GCNConv	250	2	300	447601	Adagrad	---	50	0.629370629	0.585618781
GAT_dropout	3	GATConv	350	2	350	1672301	Adagrad	---	50	0.625454545	0.586148297
GCN_3_dropout	3	GCNConv	250	2	350	347451	Adagrad	---	64	0.623188406	0.583081328
GCN_3	3	GCNConv	350	2	300	521051	Adagrad	---	64	0.619718310	0.575538325
GAT	3	GATConv	250	2	300	919601	Adagrad	---	64	0.616438356	0.56902849
GCN_3_3pools	3	GCNConv	300	2	350	551801	Adagrad	---	100	0.610738255	0.560948932
GCN_2	3	GCNConv	300	2	300	597001	Adagrad	---	64	0.604477612	0.566667187
GAT_TopK_dropout	3	GATConv	250	2	300	920351	Adagrad	---	64	0.567164179	0.524716984
GAT_TopK	3	GATConv	250	2	300	920351	Adagrad	---	64	0.563909774	0.522363134
GCN	4	GCNConv	200	2	200	318201	Adam	0.0005	64	0.561797753	0.51932728
GCN_3_3pools_dropout	3	GCNConv	300	2	300	506701	Adagrad	---	64	0.558823529	0.512928928
GCN_dropout	4	GCNConv	300	2	300	687301	Adam	0.0006	64	0.545454545	0.496258584
GCN_TopK_dropout	3	GCNConv	300	2	300	417601	Adagrad	---	64	0.536585366	0.508817398
GCN_TopK	3	GCNConv	250	2	300	323101	Adam	0.0006	64	0.505836576	0.463638064

Table 2. GNN Models and Parameters. This table shows the best model parameters of different GNN models. The top 3 GNN models (highlighted), chosen by their F1 scores, were used for the final experiment.

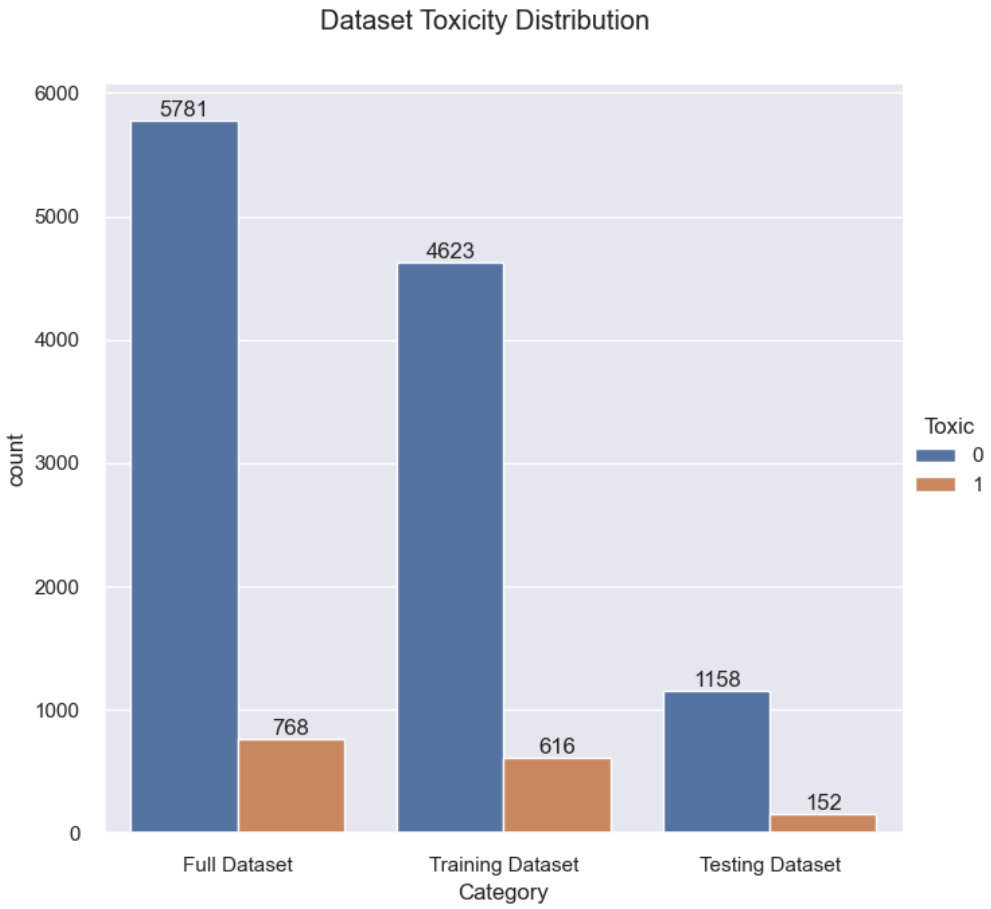

Figure 1. Dataset Toxicity Distribution. This figure shows toxicity distribution for 3 datasets: the full dataset, the training dataset (first 80% of full dataset), and testing dataset (last 20% of full dataset). Two bars are shown for each dataset, with the blue bar representing the number of nontoxic molecules and orange bar representing the number of toxic molecules, in Tox21 dataset for NR-AhR (Nuclear Receptor - Aryl hydrocarbon Receptor) target.

<table>
<tr><th>Model Name</th><th>Confusion Matrix</th></tr>
<tr><td>SVM</td><td>
<table>
<tr><td></td><td>Predicted Toxic</td><td>Predicted Non-Toxic</td></tr>
<tr><td>Toxic</td><td>102</td><td>50</td></tr>
<tr><td>Non-Toxic</td><td>99</td><td>1059</td></tr>
</table>
</td></tr>
<tr><td>GCN_2_dropout</td><td>
<table>
<tr><td></td><td>Predicted Toxic</td><td>Predicted Non-Toxic</td></tr>
<tr><td>Toxic</td><td>90</td><td>62</td></tr>
<tr><td>Non-Toxic</td><td>44</td><td>1114</td></tr>
</table>
</td></tr>
<tr><td>GAT_dropout</td><td>
<table>
<tr><td></td><td>Predicted Toxic</td><td>Predicted Non-Toxic</td></tr>
<tr><td>Toxic</td><td>86</td><td>66</td></tr>
<tr><td>Non-Toxic</td><td>37</td><td>1121</td></tr>
</table>
</td></tr>
<tr><td>GCN_3_dropout</td><td>
<table>
<tr><td></td><td>Predicted Toxic</td><td>Predicted Non-Toxic</td></tr>
<tr><td>Toxic</td><td>86</td><td>66</td></tr>
<tr><td>Non-Toxic</td><td>38</td><td>1120</td></tr>
</table>
</td></tr>
</table>

Table 3. Confusion Matrices for best SVM and 3 best GNN models. This figure shows confusion matrices (with TP, FP, FN, TN values) for SVM and 3 GNN models.

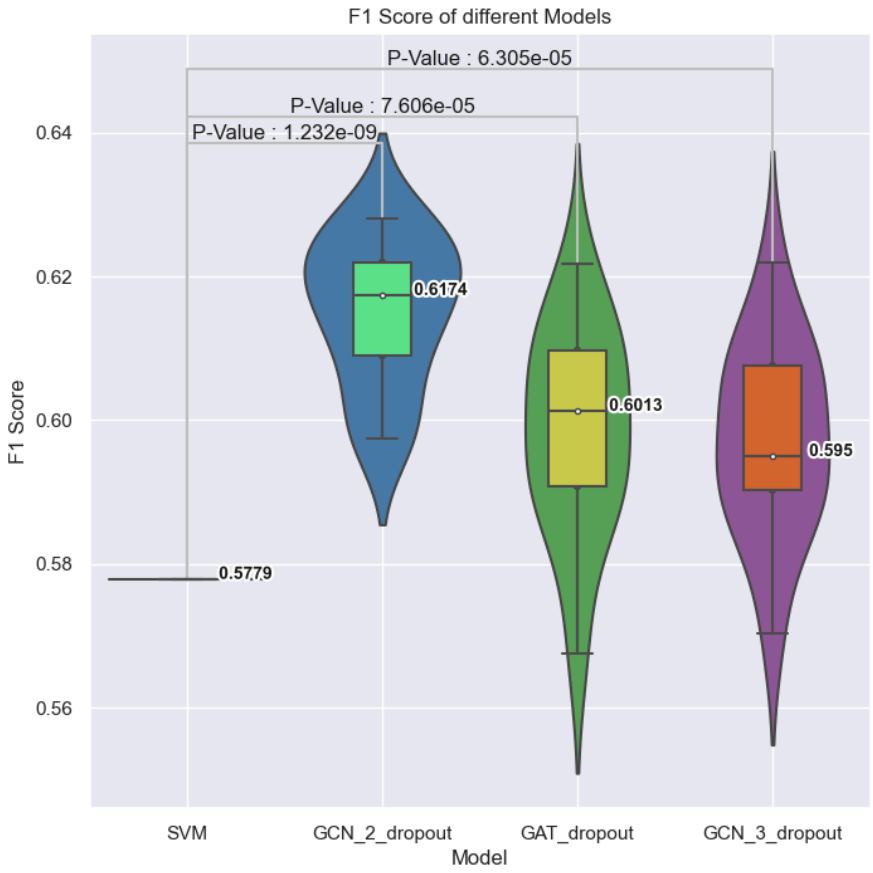

Figure 2. F1 Scores of all models and T-test P values. This figure shows F1 scores of different models, as well as T-test P values between the SVM model and each of the GNN models.

Model	Average F1 Score	95% Confidence Interval	Average Model F1 Score − Reference F1 Score (% increase)	Two tailed Welch T-test P value (between SVM and Model)
SVM	0.5779	0.5779 - 0.5779	N/A	N/A
GCN_2_dropout	0.6151	0.6094 - 0.6207	0.0372 (6.44%)	1.2315E-09
GAT_dropout	0.5983	0.5904 - 0.6063	0.0204 (3.53%)	7.6056E-05
GCN_3_dropout	0.5972	0.5898 - 0.6046	0.0193 (3.34%)	6.3053E-05

Table 4. F1 Score Average, 95% Confidence Interval and T-test P value (SVM vs Model). This table shows the average F1 score, 95% confidence interval, and T-test P value for SVM and 3 GNN models. The T-test P values are calculated between the SVM model and each of the GNN models.

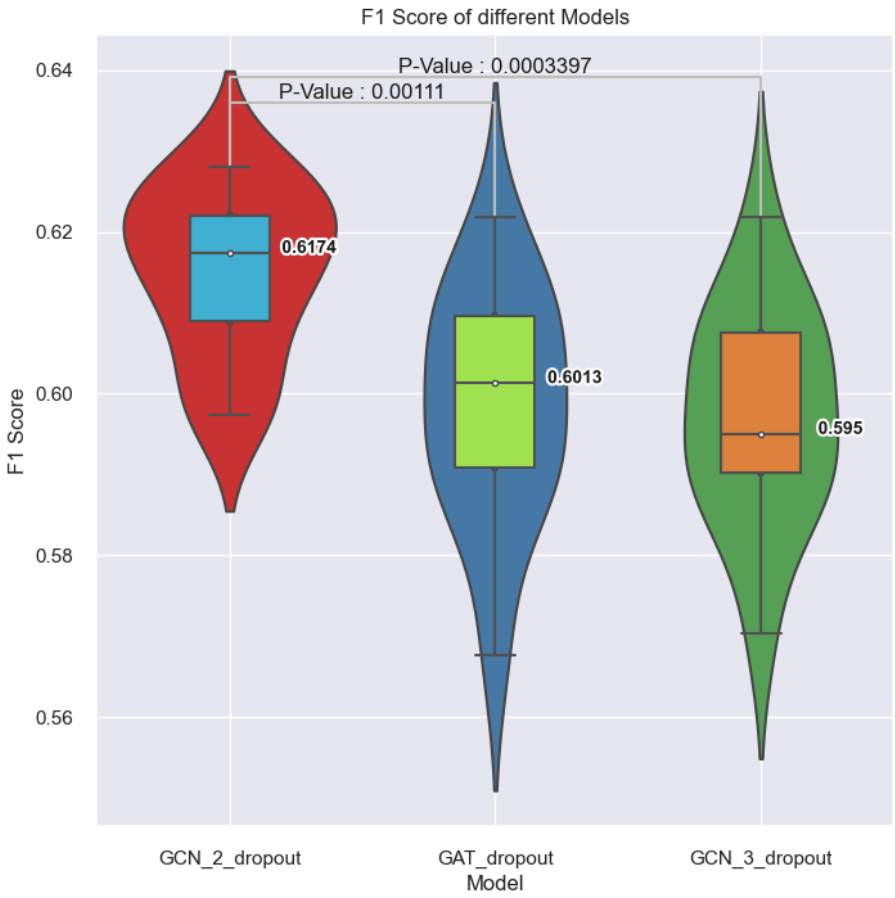

Figure 3. F1 scores of all GNN models and T-test P values. This figure shows F1-scores of different GNN models, as well as T-test P values between GCN_2_dropout and the other GNN models.

Model	Average F1 Score	95% Confidence Interval	Average Model F1 Score − Reference F1 Score	Two tailed Welch T-test P value (between GCN_2_dropout and Model)
GCN_2_dropout	0.6151	0.6094 - 0.6207	N/A	N/A
GAT_dropout	0.5983	0.5904 - 0.6063	-0.0167	0.00111
GCN_3_dropout	0.5972	0.5898 - 0.6046	-0.0178	0.00034

Table 5. F1 Score Average, 95% Confidence Interval and T-test P value (GCN_2_dropout vs Model). This table shows the average F1 score, 95% confidence interval, and T-test P value for SVM and 3 GNN models. The T-test P values are calculated between GCN_2_dropout and each of the other 2 GNN models.

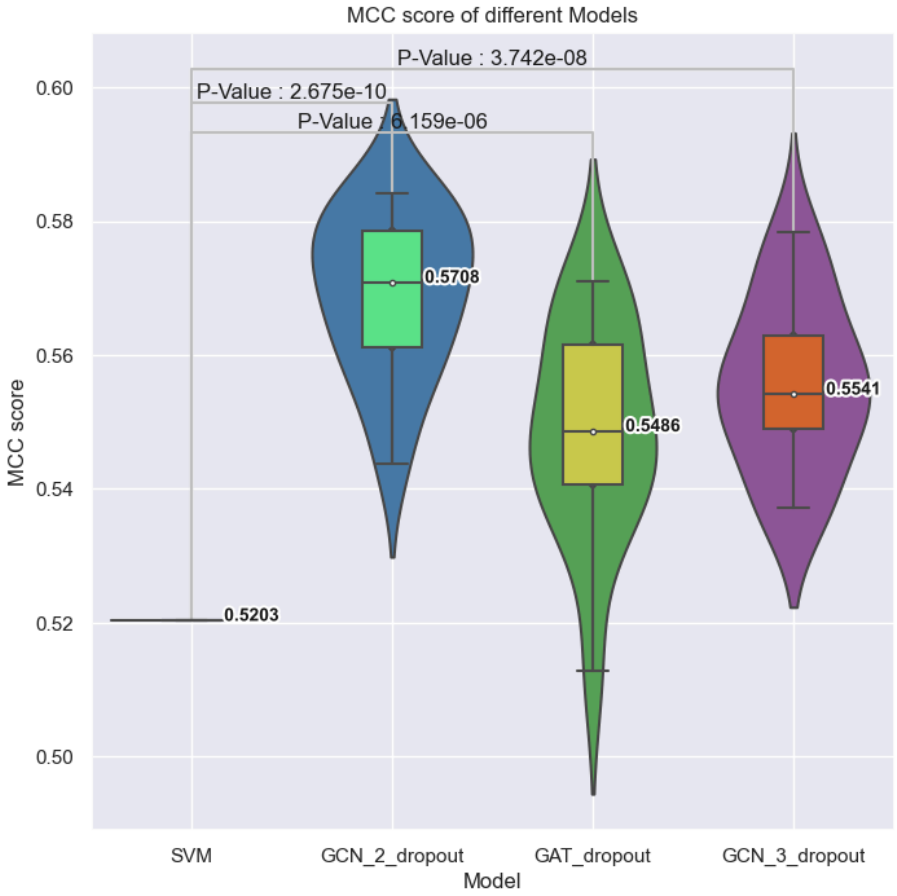

Figure 4. MCC Values of all models and T-test P values. This figure shows MCC values of different models, as well as T-test P values between the SVM model and each of the GNN models.

Model	Average MCC Value	95% Confidence Interval	Average Model MCC Value – Reference MCC Value (% increase)	Two tailed Welch T-test P value (between SVM and Model)
SVM	0.5203	0.5203 - 0.5203	N/A	N/A
GCN_2_dropout	0.5692	0.5625 - 0.5759	0.0489 (9.40%)	2.6753E-10
GAT_dropout	0.5488	0.5401 - 0.5575	0.0285 (5.48%)	6.1591E-06
GCN_3_dropout	0.5556	0.5486 - 0.5626	0.0353 (6.78%)	3.7420E-08

Table 6. MCC Value Average, 95% Confidence Interval and T-test P value (SVM vs Model). This table shows the average MCC value, 95% confidence interval, and T-test P value for SVM and 3 GNN models. The T-test P values are calculated between the SVM model and each of the GNN models.

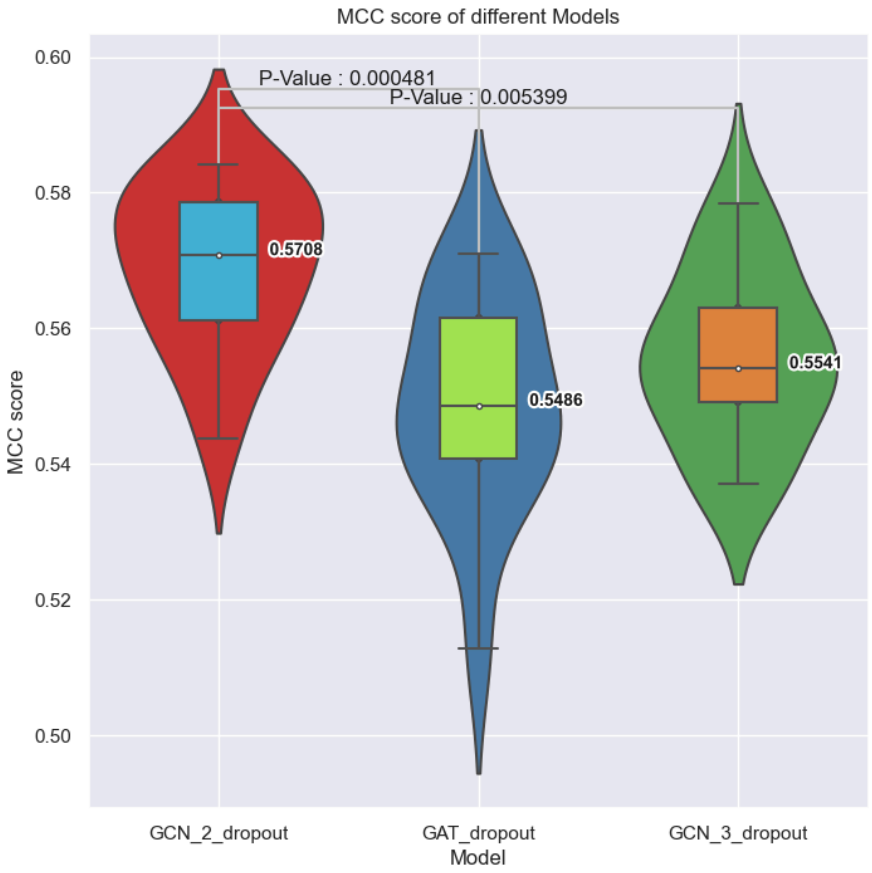

Figure 5. MCC Values of all GNN models and T-test P values. This figure shows MCC values of different GNN models, as well as T-test P values between GCN_2_dropout and the other GNN models.

Model	Average MCC Value	95% Confidence Interval	Average Model MCC Value – Reference MCC Value	Two tailed Welch T-test P value (between GCN_2_dropout and Model)
GCN_2_dropout	0.5692	0.5625 - 0.5759	N/A	N/A
GAT_dropout	0.5488	0.5401 - 0.5575	-0.0204	0.00048
GCN_3_dropout	0.5556	0.5486 - 0.5626	-0.0136	0.00540

Table 7. MCC Value Average, 95% Confidence Interval and T-test P value (GCN_2_dropout vs Model). This table shows the average MCC value, 95% confidence interval, and T-test P value for SVM and 3 GNN models. The T-test P values are calculated between GCN_2_dropout and each of the other 2 GNN models.

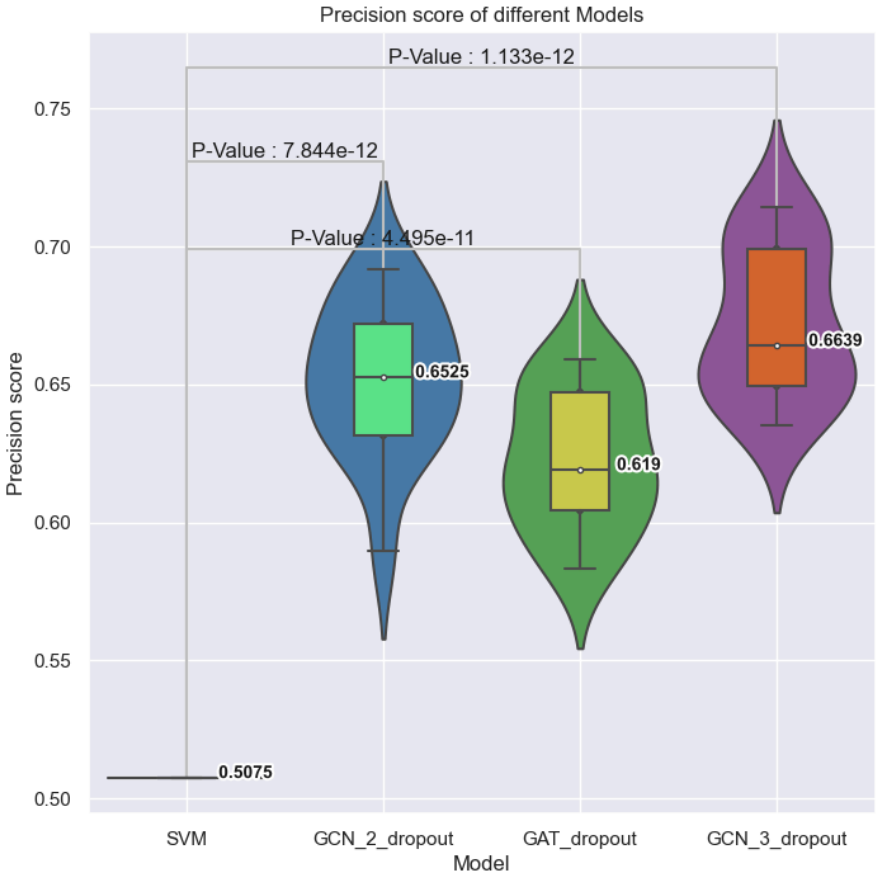

Figure 6. Precision Scores of all GNN models and T-test P values. This figure shows precision scores of different models, as well as T-test P values between the SVM model and each of the GNN models.

Model	Average Precision Score	95% Confidence Interval	Average Model Precision Score - Reference Precision Score (% increase)	Two tailed Welch T-test P value (between SVM and Model)
SVM	0.5075	0.5075 - 0.5075	N/A	N/A
GCN_2_dropout	0.6521	0.6369 - 0.6672	0.1446 (28.49%)	7.8438E-12
GAT_dropout	0.6228	0.6090 - 0.6365	0.1153 (22.72%)	4.4947E-11
GCN_3_dropout	0.6720	0.6571 - 0.6870	0.1646 (32.43%)	1.1333E-12

Table 8. Precision Score Average, 95% Confidence Interval and T-test P value (SVM vs Model). This table shows the average precision score, 95% confidence interval, and T-test P value for SVM and 3 GNN models. The T-test P values are calculated between the SVM model and each of the GNN models.

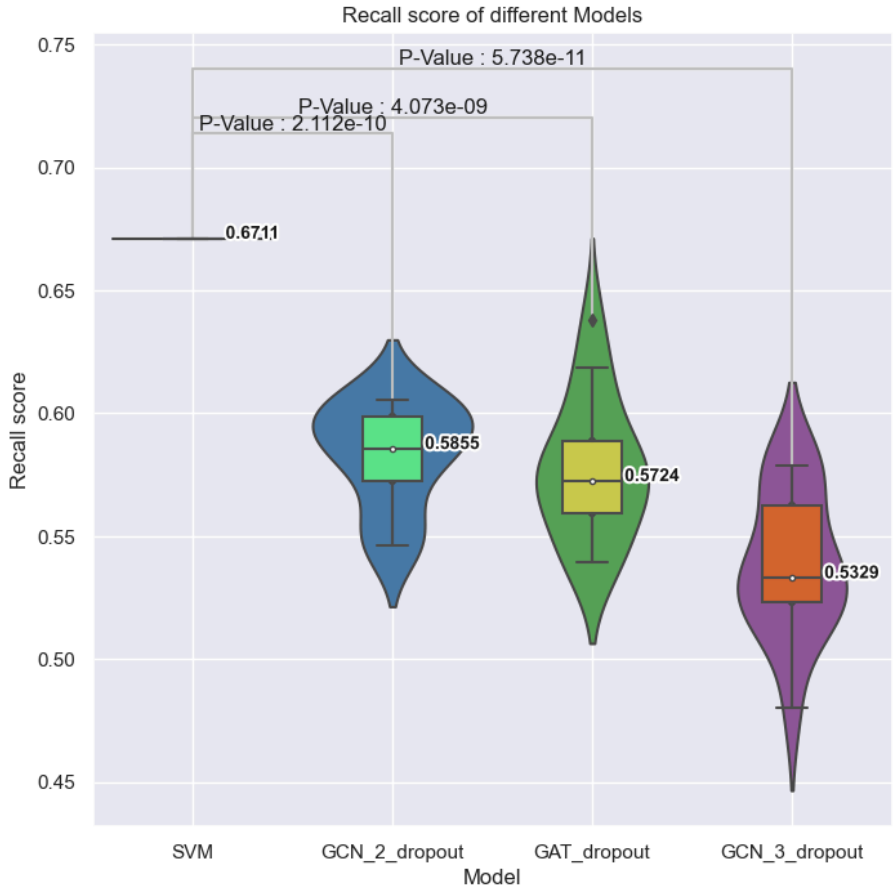

Figure 7. Recall Scores of all GNN models and T-test P values. This figure shows recall scores of different models, as well as T-test P values between the SVM model and each of the GNN models.

Model	Average Recall Score	95% Confidence Interval	Average Model Recall Score – Reference Recall Score (% decrease)	Two tailed Welch T-test P value (between SVM and Model)
SVM	0.6711	0.6711 - 0.6711	N/A	N/A
GCN_2_dropout	0.5833	0.5716 - 0.5651	-0.0877 (-13.07%)	2.1117E-10
GAT_dropout	0.5772	0.5615 - 0.5930	-0.0939 (-13.99%)	4.0727E-09
GCN_3_dropout	0.5390	0.5230 - 0.5551	-0.1320 (-19.67%)	5.7385E-11

Table 9. Recall Score Average, 95% Confidence Interval and T-test P value (SVM vs Model). This table shows the average recall score, 95% confidence interval, and T-test P value for SVM and 3 GNN models. The T-test P values are calculated between the SVM model and each of the GNN models.

DISCUSSION

As shown in Table 1, SVM model with RBF kernel has a better F1-score and MCC value than a model with a Linear kernel. Therefore, the SVM model with RBF kernel was chosen as the reference model for the final experiment. Multiple types of GNN models were created and trained using different parameters, and tested on the training dataset. The best performing models of each type were then selected, sorted by their F1-score, and compiled into Table 2, and the three best performing models (GCN_2_dropout, GAT_dropout, GCN_3_dropout) were chosen for the final experiment. Figure 1 shows toxicity information for 3 datasets: the Tox21 dataset (NR-AhR target), and 2 datasets that were created by splitting up the full dataset in 80/20 ratio. This figure explicitly shows the unbalanced datasets; the full dataset has 5781 nontoxic data but only 768 toxic data. The NR-AhR target was chosen because it was one of the least unbalanced of all target data (Tox21 contains toxicity information for 12 targets).

Different GNN models were statistically compared to the reference SVM model for F1-score, MCC value, precision score, and recall score, to see how their performance compared with the reference SVM model. GNN models were also compared with the GCN_2_dropout model, for F1-score, and MCC value, to see which GNN model had the best performance in each category. Two tailed Welch T-test was used to get P value to determine whether the results were statistically different.

SVM model had the lowest average F1-score of 0.5779. The GCN_2_dropout model had average F1-score of 0.6151, GAT_dropout had average F1-score of 0.5983 and GCN_3_dropout had average F1-score of 0.5972. T-test performed between GNN models and the SVM model gave P values of 1.2315E-09 for GCN_2_dropout, 7.6056E-05 for GAT_dropout, and 6.3053E-05 for GCN_3_dropout. Since all 3 P values are much lower than 0.0001, average F1-scores of 3 GNN models are statistically different from the SVM model. Therefore, all 3 GNN models had better F1-scores than the SVM model. Out of 3 GNN models, GCN_2_dropout had the highest average F1 score of 0.6151. When compared to GCN_2_dropout, GAT_dropout model's F1-score was 0.0167 lower (P value of 0.00111) and GCN_3_dropout model's F1-score was 0.0178 lower (P value of 0.00034). Since both P values are lower than 0.01, the average F1-score of each GNN model is statistically different from GCN_2_dropout. Therefore, GCN_2_dropout had the best F1-score among all GNN models.

The SVM model had the lowest average MCC value of 0.5203. The GCN_2_dropout model had an average MCC value of 0.5692. GAT_dropout had an average MCC value of 0.5488, while GCN_3_dropout had an average MCC value of 0.5556. GCN_2_dropout had an average MCC value that was 0.0489 higher than SVM, and P value of 2.6753E-10. GAT_dropout had an average MCC value that was 0.0285 more than SVM, with P value of 6.1591E-06. GCN_3_dropout had an average MCC value that was 0.0353 higher than SVM, with P value of 3.7420E-08. All 3 P values were significantly higher than 0.0001, so MCC value is statistically different between each model and SVM. Therefore, all 3 GNN models had better MCC values than the SVM model. Out of all 3 GNN models, GCN_2_dropout had the highest average MCC value of 0.5692. When compared to GCN_2_dropout, GAT_dropout model's average MCC value was 0.0204 lower (P value of 0.00048), and GCN_3_dropout model's average MCC value was 0.0136 lower (P value of 0.00540). Since both P values are lower than 0.01, the average

F1-score of each GNN model is statistically different from GCN_2_dropout. Therefore, GCN_2_dropout had the best MCC value among all GNN models.

The SVM model had the lowest average precision score of 0.5075. The GCN_2_dropout model had an average precision score of 0.6521. GAT_dropout had an average precision score of 0.6228, while GCN_3_dropout had an average precision score of 0.6720. GCN_2_dropout had an average precision score that was 0.1446 higher than SVM, and P value of 7.8438E-12. GAT_dropout had an average precision score that was 0.1153 more than SVM, with P value of 4.4947E-12. GCN_3_dropout had an average precision score that was 0.1646 higher than SVM, with P value of 1.1333E-12. All 3 P values were significantly higher than 0.0001, so precision score is statistically different between each model and SVM. Therefore, all 3 GNN models had better precision scores than the SVM model.

The SVM model had the highest average recall score of 0.6711. The GCN_2_dropout model had an average recall score of 0.5833. GAT_dropout had an average recall score of 0.5772, while GCN_3-dropout had an average recall score of 0.5390. GCN_2_dropout had an average recall score that was 0.0877 lower than SVM, and P value of 2.1117E-10. GAT_dropout had an average recall score that was 0.0939 less than SVM, with P value of 4.0727E-09. GCN_3_dropout had an average recall score that was 0.1320 higher than SVM, with P value of 5.7385E-11. All 3 P values were significantly higher than 0.0001, so the recall score is statistically different between each model and SVM. Therefore, SVM had a higher recall score than all 3 GNN models.

When comparing all models (SVM, GCN_2_dropout, GAT_dropout, GCN_3_dropout), GCN_2_dropout had the best F1-score and MCC value. Therefore, this project supports the hypothesis that a trained GNN model can perform better than an SVM model.

CONCLUSION

Drug toxicity prediction is one of the most crucial steps in the drug development process, as it is a measure of the safety and effectiveness of the drug. Toxicity research costs the drug industry billions of dollars, along with 10 to 15 years of research and preclinical/clinical trials, which use animal models and human testing. However, only 12% of all drugs end up being considered by the FDA for production, since many potential drugs are toxic and can no longer be pursued. By the time toxicity has been detected, an abundance of time, resources, and money have been invested into the drug, so late identification of toxicity in the drug development process can lead to the loss of millions of dollars and years of research. Therefore, it is imperative to have early detection of drug toxicity. Artificial Intelligence-based QSAR methods have been used to study relationships between molecules' chemical structures and their biological activities (e.g. k-nearest neighbors, Naive bayes, Random forest, Support Vector Machine (SVM) and Neural network (deep neural networks)). Graph Neural Networks (GNNs) are well suited to work on molecular structures, as they are designed to work on graph structures (nodes and edges). The primary objective of this computer science project was to create GNN models which can quickly and effectively predict if a molecule is toxic or not. The project compared these different GNN models with a SVM based classifier using F1-score and MCC value (Matthews Correlation Coefficient), to determine which model had better performance. The hypothesis stated that

trained GNN models should have better F1-score and better MCC value than the best SVM model.

This project was done in VSCode, using a Python notebook. The dataset used in this project, Tox21, was imported into the notebook using MoleculeNet() function of PyTorch Geometric. This dataset contained 6549 molecules with 12 targets of toxicity, among other information. Most of the targets had highly imbalanced toxicity data, as there were many more nontoxic molecules than toxic molecules. For this experiment, NR-AhR target was used because it was one of the least imbalanced of all target data. For this target, the dataset contained 5781 nontoxic molecules but only 768 toxic molecules. RDKit Fingerprints were also added to the dataset (for SVM training and evaluation), and atomic features were hot encoded in an array of 184 bits (for GNN model training and evaluation). Full dataset was then split into training and testing dataset in 80/20 ratio. This was to ensure that both SVM and GNN models were trained and evaluated on the same molecular dataset. Once the datasets were prepared, multiple GNN models were written, using different numbers and types of GCN layers, activation functions and pooling layers. These models were then trained using different parameters (different number of hidden GCN features, hidden Linear features, optimizer, learning rates, number of epochs, batch sizes) to see which combination resulted in the highest F1-score for each model. F1-score and MCC value was used to compare models' performance, because it is a better metric when dealing with unbalanced datasets, as compared to precision and recall scores. All models were sorted by F1-score, and the top 3 GNN models (GCN_2_dropout, GAT_dropout, GCN_3_dropout) were chosen for the final experiment, along with their parameters. Similarly, multiple SVM models were created, using different C values, kernels, gamma, and class weights. The model with highest F1-score was chosen for the final experiment, along with its parameters, to be the reference model to gauge GNN model performance. These 4 best models were then trained and evaluated 15 times, and data collected was then used for final analysis.

FUTURE WORK

Early detection of toxicity is crucial in the drug development process in order to avoid costly, lengthy research. Most machine learning QSAR method utilize shallow algorithms (eg. Support Vector Machines (SVMs), k-Nearest neighbors (K-NN), random forest). Some deep learning models have shown better results than these shallow algorithms, but those models don't take into account molecular graph structures information in their models (DeepTox). This project shows how a Graph Neural Network (GNN) based deep learning model can make better predictions than shallow algorithms. This opens the possibilities of creating more complex deep learning models by combining traditional deep learning methods (molecular feature information, such as molecular fingerprints) with GNNs (molecular structure information). Another potential research area will be to utilize intermediate layer data with SVM and see if this produces better predictions. This project only explored two types of GNNs (GCNConv and GATConv). Further research can explore the utilization of other types of GNNs, to see if they have better performance (F1-score). Even though this project was successful in predicting molecular toxicity, it lacked explainability of molecular toxicity. It did not answer "why" some molecules are toxic while others are non-toxic, or provide models that can detect patterns between molecules that are toxic or non-toxic. Further research is needed to find out if a model can answer the question of "why" this occurs.

REFERENCES

1. AnaConda. (n.d.). *Free Download*. Anaconda. Retrieved September 24, 2023, from https://www.anaconda.com/download
2. Britannica. (2023, September 21). *Artificial Intelligence*. Britannica. Retrieved September 21, 2023, from https://www.britannica.com/technology/artificial-intelligence
3. Brown, S. (2021, April 21). *Machine learning, explained*. MIT Management Sloane School. Retrieved September 21, 2023, from https://mitsloan.mit.edu/ideas-made-to-matter/machine-learning-explained
4. Columbia Mailman School of Public Health. (2020, November 30). *What is Toxicology?* Columbia Mailman School of Public Health. Retrieved September 17, 2023, from https://www.publichealth.columbia.edu/news/what-toxicology
5. Dorato, M. A., & Buckley, L. A. (2007). Toxicology testing in drug discovery and development. *NIH PubMed*, *31*(1). https://doi.org/10.1002/0471141755.tx1901s31
6. Food and Drug Administration [FDA]. (n.d.). *The Drug Development Process*. U.S. Food and Drug Administration. Retrieved September 17, 2023, from https://www.fda.gov/patients/learn-about-drug-and-device-approvals/drug-development-process
7. GitHub. (n.d.). *JupyterLab-Desktop*. Github. Retrieved September 24, 2023, from https://github.com/jupyterlab/jupyterlab-desktop
8. Guengerich, F. P. (2010). Mechanisms of drug toxicity and relevance to pharmaceutical development. *NIH National Library of Medicine*, *26*(1), 3-14. https://doi.org/10.2133%2Fdmpk.dmpk-10-rv-062
9. Guha, R. (2013). On exploring structure–activity relationships. *Methods in Molecular Biology*, 81-94. https://doi.org/10.1007/978-1-62703-342-8_6
10. IBM. (2021, March 8). *Support Vector Machine Models.* IBM. Retrieved October 7, 2023, from https://www.ibm.com/docs/en/spss-modeler/18.1.0?topic=nodes-support-vector-machine-models
11. IBM. (n.d. a). *What are convolutional neural networks?* IBM. Retrieved September 24, 2023, from https://www.ibm.com/topics/convolutional-neural-networks
12. IBM. (n.d. b). *What are recurrent neural networks?* IBM. Retrieved September 24, 2023, from https://www.ibm.com/topics/recurrent-neural-networks
13. IBM. (n.d. c). *What is a neural network?* IBM. Retrieved September 21, 2023, from https://www.ibm.com/topics/neural-networks
14. K, B. (2020, December 22). *Everything You Need To Know About Jupyter Notebooks*. Towards Data Science. Retrieved September 24, 2023, from https://towardsdatascience.com/everything-you-need-to-know-about-jupyter-notebooks-10770719952b
15. Labonne, M. (2023). *Hands-On Graph Neural Networks Using Python*. Packt Publishing. *THE LEADING TEAMS' AUC RESULTS ON THE FINAL TEST SET IN THE TOX21 CHALLENGE* [Image]. (n.d.). https://www.frontiersin.org/articles/10.3389/fenvs.2015.00080/full
16. Lo, Y.-C., Rensi, S. E., Torng, W., & Altman, R. B. (2018). Machine learning in chemoinformatics and drug discovery. *Drug Discovery Today*, *23*(8), 1538-1546. https://doi.org/10.1016/j.drudis.2018.05.010

17. MatchTrial. (2020, August 25). *How long does it take to develop a new drug?* MatchTrial. Retrieved September 15, 2023, from https://matchtrial.health/en/how-long-does-it-take-to-develop-a-new-drug/#
18. Mayr, A., Klambauer, G., Unterthiner, T., & Hochreiter, S. (n.d.). DeepTox: Toxicity Prediction using Deep Learning. *Frontiers*. https://www.frontiersin.org/articles/10.3389/fenvs.2015.00080/full
19. Menzli, A. (2023, September 11). *Graph Neural Network and Some of GNN Applications: Everything You Need to Know*. Neptune. Retrieved September 21, 2023, from https://neptune.ai/blog/graph-neural-network-and-some-of-gnn-applications
20. NVIDIA. (n.d.). *PyTorch*. Nvidia. Retrieved September 24, 2023, from https://www.nvidia.com/en-us/glossary/data-science/pytorch/
21. OECD. (n.d.). *Introduction to (Quantitative) Structure Activity Relationships*. OECD. Retrieved September 19, 2023, from https://www.oecd.org/env/ehs/risk-assessment/introductiontoquantitativestructureactivityrelationships.htm#:~:text=Structure%2DActivity%20Relationship%20(SAR),target%20property)%20of%20studied%20compounds
22. Pelikan, E. W. (2022). *Glossary of Terms and Symbols Used in Pharmacology*. Boston University. Retrieved September 17, 2023, from https://www.bumc.bu.edu/busm-pm/resources-2/glossary/#d
23. PyG. (n.d.). *PyG Documentation*. PyTorch Geometric (PyG). Retrieved September 24, 2023, from https://pytorch-geometric.readthedocs.io/en/latest/
24. PyTorch. (n.d.). *PyTorch*. PyTorch. Retrieved September 24, 2023, from https://pytorch.org/
25. *RDKit: Open-Source Cheminformatics Software*. (n.d.). RDKit. Retrieved September 24, 2023, from https://www.rdkit.org/
26. Sanchez-Lengeling, B., Reif, E., Pearce, A., & Wiltschko, A. B. (2021, September 2). *A Gentle Introduction to Graph Neural Networks*. Distill. Retrieved September 21, 2023, from https://distill.pub/2021/gnn-intro/
27. *Tox21 Data Challenge 2014*. (2014). NIH Tox21. Retrieved September 24, 2023, from https://tripod.nih.gov/tox21/challenge/about.jsp
28. Verboon, C. (Ed.). (2021, April). *Research and Development in the Pharmaceutical Industry*. Congressional Budget Office. Retrieved September 15, 2023, from https://www.cbo.gov/publication/57126
29. Visual Studio Code. (n.d.). [Code editing. Redefined. Free. Built on open source. Runs everywhere.]. Visual Studio Code. Retrieved September 24, 2023, from https://code.visualstudio.com/
30. Western Governors University [WGU]. (2020, May 26). *What is deep learning?* Western Governors University. Retrieved September 21, 2023, from https://www.wgu.edu/blog/what-deep-learning2005.html