# Deep Drug Repurposer

Sarthak Kothari, Noah Demoes, Omair Shafi Ahmed, Forrest Hooton

## Summary

In 2017, the international pharmaceutical industry generated over USD 934 billion in revenue and is expected to grow to over $1,170 billion in 2021. [1] Although people are willing to pay for pharmaceuticals, the cost associated with creating new, breakthrough drugs are costly and often unsuccessful. Only 12% of drugs that are in clinical phase will be approved for market. [4] Drug development for a single drug costs USD 2.6 billion and 17 years to develop on average. [4] Development regulations drive over 10 percent of this cost, averaging USD 339.3 million dollars. The U.S. Food and Drug Administration (FDA) requires three phases of clinical testing: determining human safety and dosing (phase 1), initial reading of efficacy and side effects (phase 2), and larger clinical trials to determine the safety and efficacy in larger numbers (phase 3). [3] Bypassing any stage in this process could create cost and time savings for drug development and reduce the burden on consumers.

One method that can reduce drug development cost is drug repurposement. Drug repurposement explores new applications for approved drugs or drugs that have passed at least one of the three FDA approval phases. Since these candidate drugs have already been approved by the FDA and information about pharmacology, formulation, dose, and potential toxicity is already available, it is easier for drug researchers to fulfill phase 1 and phase 2 of clinical trials. This allows a pharmaceutical firm to focus on drug efficacy in more probable drugs and save costs associated with drug development.

At the moment, drug repurposing is pursued through investigating drug side-effects to determine if the drug could be useful for other conditions. A classic example for this method is *sildenafil*, which was originally developed to treat hypertension. The drug also had side effects that could be used to treat another disease - erectile dysfunction in men. Sildenafil was repurposed to treat erectile dysfunction, and is now well known as Viagra. However, the intense manual analysis required for the drug side-effect investigation strategy limits the procedure for broad drug development. We propose a better way to automatically predict a drug and disease target interaction potential.

Our project aims to develop a deep learning model and web portal for drug repurposing. Specifically, we seek to use information about the molecular structure of drugs and their target receptors to predict whether or not the two inputs react, i.e. a binary classifier. A breakthrough in drug-receptor repurposing could bypass toxicity stages in drug trials, and save tens of millions of dollars for a single effort to treat a disease. There are other active efforts for drug-repurposing, however these efforts use limited datasets that comprise a small fraction of

chemical-target reactions [9]. We plan to couple deep learning techniques with a more robust dataset to develop our drug structure based receptor model. Our primary datasets will be DrugBank for model development, PubChem for chemical metadata, and the BindingDB database as a validation dataset [12][6][15].

# Methods

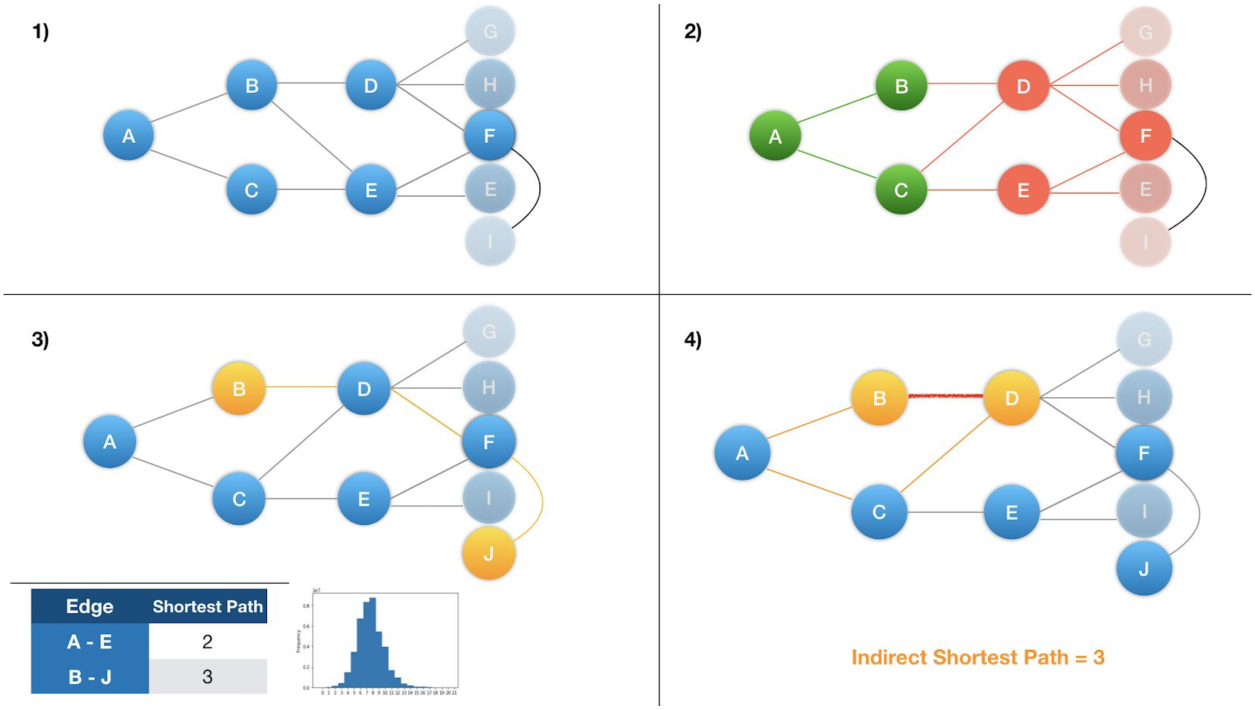
## Data Curation and Transformation

The first step is data curation and transformation. DrugBank outputs its database as a large xml data dump. The DrugBank dump contains information such as drug name, drug interactions, drug targets, and associated drug names. We transformed the raw data into a drug-target bipartite network where edges represent positive links between a drug and a target gene. Drugs are represented by their chemical structure in the DrugBank database. However, this representation does not enable differentiation at the machine level. For example, benzaldehyde's chemical formula is $C_6H_5CHO$, but this does not capture the bond structure that is central in drug repurposement identification. Fortunately, chemical compounds can be represented using the simplified molecular input line entry system (SMILES) that captures the bond structure in its representation. The SMILES representation is a machine readable depiction of a chemical compound which is necessary to create a machine learning based drug repurposer. For example, the SMILES representation of benzaldehyde is C1=CC=C(C=C1)C=O. We further enhance the machine readability by converting the SMILES to their respective Morgan Fingerprint [10]. Thus, we substitute the fingerprint representation to characterize each drug. In order to get the fingerprint representation for the drugs in the DrugBank database we will retrieve chemical SMILES and convert them to their Morgan Fingerprint using the RDKit python package [7].

After generating our chemical fingerprints, we selected our targets. DrugBank contains two candidate targets, proteins or genes. We decided to use target genes instead of proteins because genes are more foundational than proteins. Proteins are transcribed by genes; thus, if we create an accurate classifier for target genes it stands to reason we can identify the proteins as well; although, we are not guaranteed this ability if we decide to target proteins. Genes are presented as strings that contain some combination of the four characters A,T,G, and C. Additionally, our drugs are character strings. Because both variables are strings we can natural language processing (NLP) techniques to create fingerprint embeddings to represent each variable.

While many deep learning models choose to use one-hot encoding for their text input, we decided to attempt using embeddings. We select embedding instead of one-hot encoding to represent our text input because it enables one to quantify semantic similarity between two different text inputs and it is more computationally efficient to one-hot encoding for this problem. An embedding is a continuous, numerical representation of a categorical variable, mapped to a specific length vector. We used an embedding length of 100 for this project. The dense representation of embeddings allows us to pre-compute features of chemicals for different models. This decreases the memory requirements for models and data storage, and captures patterns for less complex models such as logistic regression. We chose to use Mol2Vec to embed the chemical molecules, and a fasttext embedding to embed genes [13]. Mol2Vec is a molecule processing method that uses chemical fingerprints with Word2Vec to encode chemical structure as a dense vector, and fasttext captured relationships between groupings of characters for genes [13].

## Negative Sampling



| Edge | Shortest Path |
|---|---|
| A - E | 2 |
| B - J | 3 |

***Figure 1:*** *1) The original graph. 2) Example of naive negative sampling for node A. All nodes in red are considered as negative samples for A. 3)  An example of all pairs shortest path - in this case B - J. 4) An example of shortest indirect path for edge B - D.*

A challenge with using DrugBank is that it only provides true (label = 1) samples, and training a binary classifier requires both true and false examples. A naive way to approach this would be to classify any drug target that does not have a direct link as negative (Fig. 1.2). The problem with this naive approach is that this list of negative samples would 1) greatly outnumber the number of positive samples, and 2) include potential positive samples as negative.

For these reasons, we attempted to pick our negative samples using a methodology that would avoid the stated pitfalls. First, we look at all pairs shortest paths (APSP) and calculate the mean shortest path for all pairs (Fig 1.3). This gives us a baseline path length for the network. Next, we look at the drug-target edges, and we calculate the second shortest path from the drug to the target for each drug-target edge (Fig. 1.4). We then pick a threshold based on the distribution of the second shortest indirect paths. All shortest bipartite paths from the APSP that cross this threshold are considered as negative sample.

## Model

The deep binary classification model we will implement is inspired from the recent developments in drug-target prediction which is WideDTA model. [14] We used 3 different variations of proposed WideDTA model for our project. Dense Model / Deep NN for Binary Classification, Common Convolution Model for Binary Classification and Converging Convolution for Binary Classification. For the Deep NN model, we provide drug-target embeddings as inputs which is passed to a series of fully connected hidden layers with different activation functions and different dropouts rate and then outputs a prediction. Similarly, the input for Common Convolution is also the drug-target embeddings, however, we convolute them using a series of convolution layers before passing it to the Deep NN. In the Converging CNN model, the drug embeddings and target embeddings are convoluted separately using a series of separate convolution layers, these convoluted embeddings are then concatenated and passed to the Deep NN structure.

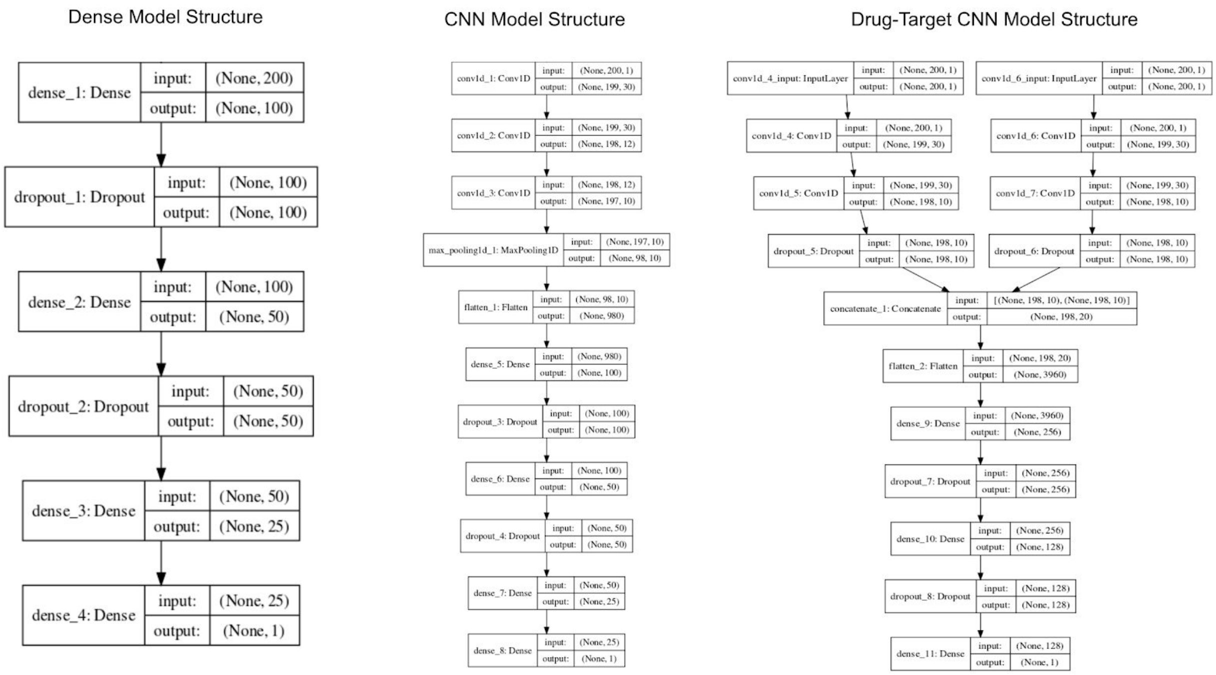

Dense Model Structure
dense_1: Dense input: (None, 200) output: (None, 100)
dropout_1: Dropout input: (None, 100) output: (None, 100)
dense_2: Dense input: (None, 100) output: (None, 50)
dropout_2: Dropout input: (None, 50) output: (None, 50)
dense_3: Dense input: (None, 50) output: (None, 25)
dense_4: Dense input: (None, 25) output: (None, 1)
CNN Model Structure
conv1d_1: Conv1D input: (None, 200, 1) output: (None, 199, 30)
conv1d_2: Conv1D input: (None, 199, 30) output: (None, 198, 12)
conv1d_3: Conv1D input: (None, 198, 12) output: (None, 197, 10)
max_pooling1d_1: MaxPooling1D input: (None, 197, 10) output: (None, 98, 10)
flatten_1: Flatten input: (None, 98, 10) output: (None, 980)
dense_5: Dense input: (None, 980) output: (None, 100)
dropout_3: Dropout input: (None, 100) output: (None, 100)
dense_6: Dense input: (None, 100) output: (None, 50)
dropout_4: Dropout input: (None, 50) output: (None, 50)
dense_7: Dense input: (None, 50) output: (None, 25)
dense_8: Dense input: (None, 25) output: (None, 1)
Drug-Target CNN Model Structure
conv1d_4_input: InputLayer input: (None, 200, 1) output: (None, 200, 1)
conv1d_6_input: InputLayer input: (None, 200, 1) output: (None, 200, 1)
conv1d_4: Conv1D input: (None, 200, 1) output: (None, 199, 30)
conv1d_6: Conv1D input: (None, 200, 1) output: (None, 199, 30)
conv1d_5: Conv1D input: (None, 199, 30) output: (None, 198, 10)
conv1d_7: Conv1D input: (None, 199, 30) output: (None, 198, 10)
dropout_5: Dropout input: (None, 198, 10) output: (None, 198, 10)
dropout_6: Dropout input: (None, 198, 10) output: (None, 198, 10)
concatenate_1: Concatenate input: [(None, 198, 10), (None, 198, 10)] output: (None, 198, 20)
flatten_2: Flatten input: (None, 198, 20) output: (None, 3960)
dense_9: Dense input: (None, 3960) output: (None, 256)
dropout_7: Dropout input: (None, 256) output: (None, 256)
dense_10: Dense input: (None, 256) output: (None, 128)
dropout_8: Dropout input: (None, 128) output: (None, 128)
dense_11: Dense input: (None, 128) output: (None, 1)

***Figure 2:** Shows the different model architecture for the Deep NN, CNN, and Converging CNN.*

# Results

Given that we plan to work on this task for both project periods of the semester, our results discuss the outcomes for project progress. This progress update observes intermediate goals that setup the remainder of our project. More specifically, we will review the outcome of negative sampling, embedding results, early modeling results, and validation data choices.

## Negative Sampling Results

After running the negative sampling technique, we end up with 231,117 examples of interactions that have very little evidence of interacting with one another. The 231k samples is based on the indirect path length cutoff of 12 hops. Most indirect shortest paths average around 3 hops and most all pairs shortest path average around 8 hops. We believe the difference between average path hops is evidence that drugs and targets in close network proximity have a higher probability of interacting with one another. Since there are no indirect path lengths greater than 11, we pick a path length of 12 as cutoff and any pair with a path length greater than 12 as a negative sample.

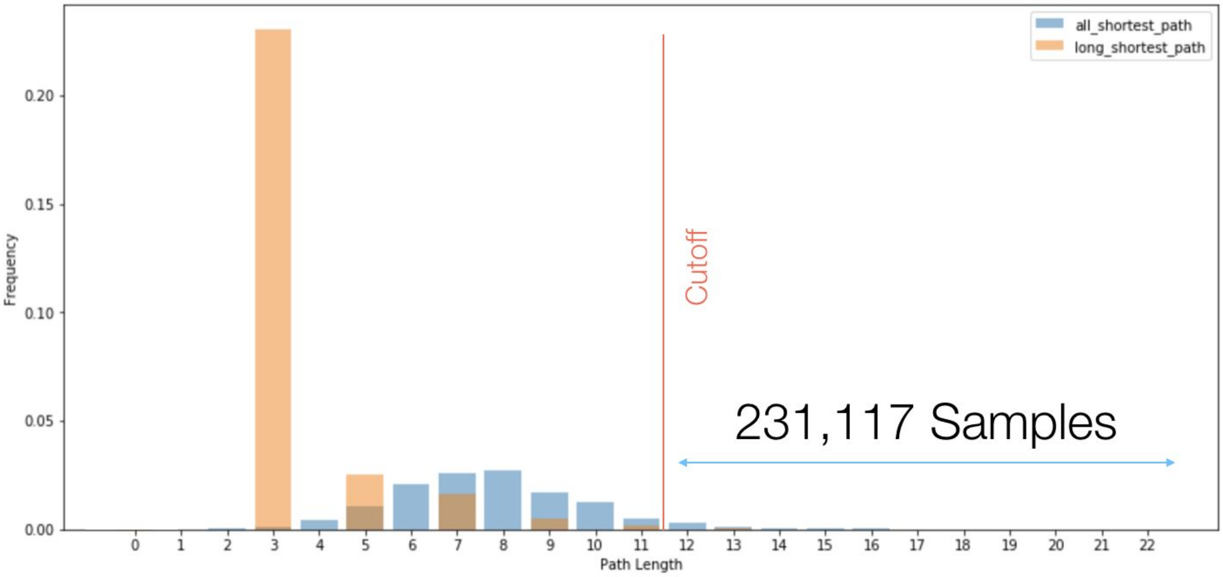


***Figure 3:*** *Shows the difference in the distributions between the all pairs shortest path of the bipartite network and the longest shortest path of the bipartite network using the technique described above.*

## Embedding Results

Although we do not have explicit metrics to measure the success of our embeddings, we applied the UMAP algorithm on the drug and gene embeddings to plot the values (Fig 4) and investigate if any meaningful differences are captured. [8] The drug embedding vectors have different clusters, and are therefore capturing some difference between the drug fingerprints (Fig 4.1). Furthermore, the colors correspond to clusters assigned on the full embedding vectors using the k-means algorithm, and before applying UMAP. We can see by the overlapping colors, specifically red, light blue, and green, that the higher dimension embedding vectors capture more complexity than shown in Fig 4.1. The gene embedding vectors seem to be less clustered with less cluster overlap, therefore capturing less difference (Fig 4.2). However, the embedding reduction is not scattered at random, and so is still capturing some differences between genes. We conclude that while the gene embeddings do not capture the level of detail as the drug embeddings, they still capture enough information to move forward with the gene embeddings as an input to our model.

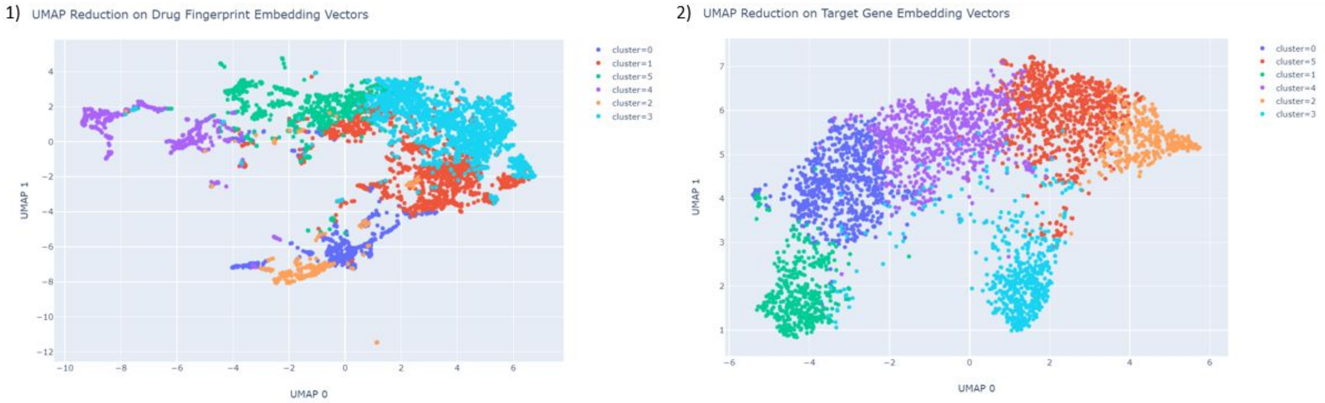


***Figure 4:*** *1) UMAP reduction of drug Mol2Vec embeddings. 2) UMAP reduction of gene fasttext embeddings.*

## Model Results

We began with a baseline logistic classification model as a well-known benchmark to help use understand the performance of more advanced models. We then developed the three deep networks as shown in Fig 2. We chose the F1-score as the validation method for our models, as the F1-score incorporates both precision and recall. Logistic model had a F1-score of ~58%, whereas the Deep NN model was performing much better with an F1 score of ~76%. Even though we achieved similar results in the common CNN model (F1-score of ~68%), it only predicted True for all samples, hence we do not currently consider it a well-performing model. Although the converging CNN performed much worse than all the other models, it classified examples as both True and False unlike the common CNN model. The higher F1-score for the DeepNN compared to logistic regression indicates non-linear boundaries. We originally anticipated higher accuracies using the CNN models than the DeepNN, however the CNNs failed to perform as per our expectations. A possible explanation is our use of embeddings as input to the CNNs, which already capture patterns in the raw data.

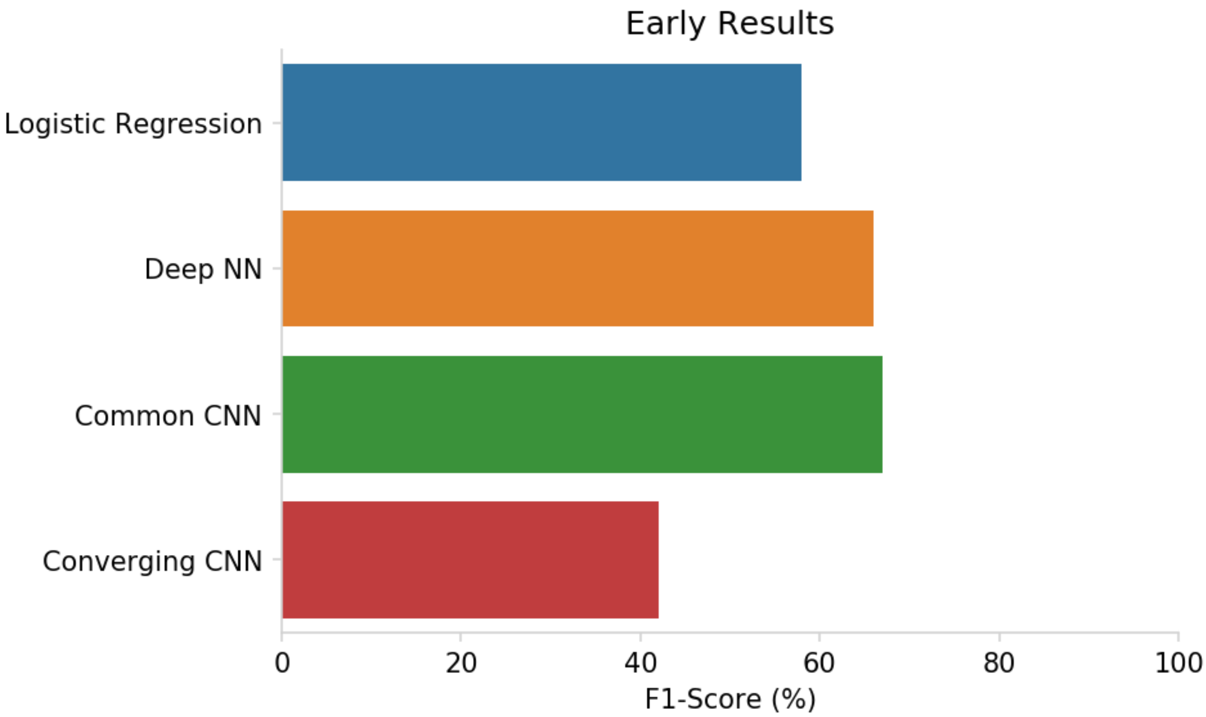


***Figure 5:*** *Shows the early F1-score results of models on the test dataset.*

## Convolutions, Embeddings and Graph Representations

An important choice we made in the project was the feature engineering concerning the gene and chemical fingerprints. With the preliminary literature review, we decided to convolve over fingerprint embeddings. However, later into the project, the theoretical underpinnings of convolving over chemical fingerprint embeddings became more unclear. To that end, we decided to dig deeper about the interaction of convolutions and embeddings and modify our model appropriately.

As shown in Fig 6 [11], a convolutional layer is used to perform a filtering operation on an image. In a deep convolutional neural network, these filters are learned over multiple examples, learning to filter high level features, in every layer of convolution, that are important for the classification task at hand, from the underlying data.

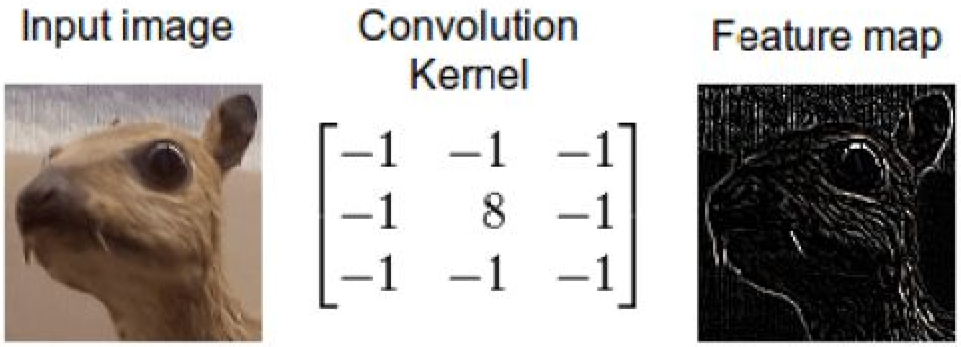


***Figure 6: An example of a convolutional filter that extracts the shape of an animal's head. This filter is then used by layers downstream to further extract higher level features, or use this directly to classify this image.***

Alternatively, word vectors use the same high level structural information of the underlying data, but instead map that information into a lower dimensional latent space. In this latent space the distance between two vectors is indicative of the similarities of their high dimensional features. This is useful in classifying the data downstream as a classifies can learn to classify images based on their distances. However, we theorize that embedding our fingerprint vectors into a high dimensional space masks the high level structural information that a convolutional layer can filter over which likely reduces the effectiveness of the convolution layer. We plan to evaluate further in part two of the project.

## Validation Data

In addition to the DrugBank testing dataset, we also want to use a validation set external to DrugBank. If the model produces similar results on the external datasets, we can be more confident that our model is capturing patterns related to chemical-gene relationships rather than only learning how to capture intricacies of DrugBank. We choose to use BindingDB because of its large number of samples and pre-listed chemical SMILES information and gene information. However, BindingDB lists many chemical-gene interactions, and does not label which reactions are actually used as drugs. Therefore, we will need to assign labels to the samples as which pairs represent drugs (samples of label 1, 'drug'), and which do not have a drug relationship (samples of label 0, 'not drug').

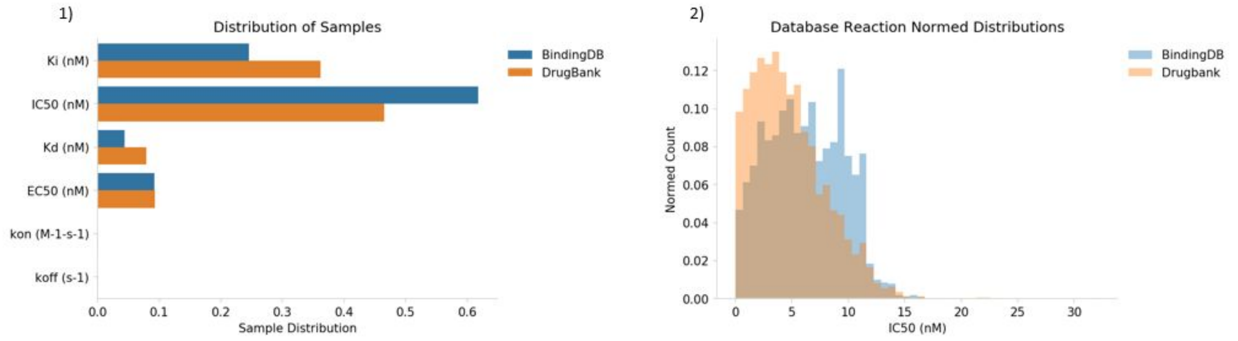


***Figure 7:*** *1) The distribution of chemical reaction measurements for all of Binding DB and listed DrugBank samples. 2) The normalized IC50 measurement distribution for all of Binding DB and listed Drugbank samples.*

We choose to create a threshold for chemical reaction measurements to create ‘drug’ and ‘not drug’ samples. The measurement units of Binding DB indicate stronger reactions as the measurement decreases, and weaker reactions as the measurement increases. A sample is labeled as a drug interaction if the chemical measurement falls below the threshold, and not a drug interaction if it falls above. We merged the BindingDB and DrugBank datasets contained matches in drug-gene interactions, and observed the distribution of the databases to choose a threshold. To complicate matters, BindingDB records chemical-gene interactions with several different types of reaction measurements. Both databases listed measurements in units of IC50 most frequently (Fig 7.1), therefore we chose to pick samples with entries using the IC50 measurement. We compared the distributions for BindingDB and DrugBank in Fig 7.2. This comparison shows a distinct dropoff for the number of drugs at an IC50 value of seven, therefore we will begin with seven as our threshold. If we use a threshold of seven, then BindingDB produces 132,004 drug relationships and 922,635 non-drug relationships.

# Discussion

We aggregated the necessary components of our drug repurposing model in part one of our project. Our objectives were to create drug and gene embeddings, compute a negative sample dataset, implement various working model structures, and develop a validation dataset. We were able to create drug and gene embeddings that captured sufficient variance for to be used as model inputs, and will move forward with the current embedding approach by incorporating the DrugBank and BindingDB validation set embeddings. We also developed 231,117 negative samples for the training dataset, and 132,004 drug relationships and 922,635 non-drug relationships from BindingDB using a thresholding approach. Our models are running, and the common Deep NN is currently the highest performing model.

One adjustment we made was to use BindingDB as the validation set instead of the papers we discussed earlier. The primary reason for the change is that BindingDB already contained molecule SMILE's and gene's, while the other sets only contained names with no guarantee that we could properly match the names in PubChem and retrieve the correct chemical structures. Furthermore, BindingDB contains many more samples than the validation databases we stated in the proposal. We could have possibly predicted the chemical reaction strength rather than doing a binary classifier of 'drug'/'not drug' because there was overlap between drug-gene reactions between BindingDB and DrugBank. However, this would have reduced the number of usable DrugBank samples, and being non-experts it would be more difficult for us to interpret the output of the model and its implications.

Moving forward, model engineering will be our primary scientific focus. Given the challenges associated with feature engineering in the Results section, we plan on implementing a model that directly convolves over the fingerprints. Furthermore, we would also like to tune the parameters of our best performing network varying the number of layers, nodes, epochs and dropouts. We anticipate that these model improvement procedures will cause our top F1-score to increase, and will deploy the best model in a web portal.

# Statement of Contributions

Forrest Hooton casted the high level vision of the project and implemented the embeddings for the drugs/targets and visualizing it using UMAP.

Noah DeMoes was responsible for embedding the drugs/targets while also experimenting with the model.

Sarthak Kothari was responsible for building out the models and debugging them.

Omair Shafi Ahmed was responsible for implementing the negative sampling strategy.

# References

[1] Business Research Company. "The Growing Pharmaceuticals Market: Expert Forecasts and Analysis." *Market Research Blog*, blog.marketresearch.com/the-growing-pharmaceuticals-market-expert-forecasts-and-analysis.

[2] Ciallella, John, and John Ciallella. "A New Use for Old Drugs." *Eureka*, 7 Aug. 2019, eureka.criver.com/a-new-use-for-old-drugs/.

[3] Commissioner, Office of the. "Step 3: Clinical Research." *U.S. Food and Drug Administration*, FDA, www.fda.gov/patients/drug-development-process/step-3-clinical-research.

[4] Dimasi, Joseph A., et al. "Innovation in the Pharmaceutical Industry: New Estimates of R&D Costs." *Journal of Health Economics*, vol. 47, 2016, pp. 20–33., doi:10.1016/j.jhealeco.2016.01.012.

[5] Jaeger, Sabrina, et al. “Mol2vec: Unsupervised Machine Learning Approach with Chemical Intuition.” *Journal of Chemical Information and Modeling*, vol. 58, no. 1, 2018, pp. 27–35., doi:10.1021/acs.jcim.7b00616.

[6] Kim, Sunghwan, et al. “PubChem 2019 Update: Improved Access to Chemical Data.” *Nucleic Acids Research*, vol. 47, no. D1, 2018, doi:10.1093/nar/gky1033.

[7] Landrum, greg. “Open-Source Cheminformatics Software.” *RDKit*, www.rdkit.org/.

[8] Mcinnes, Leland, et al. “UMAP: Uniform Manifold Approximation and Projection.” *Journal of Open Source Software*, vol. 3, no. 29, 2018, p. 861., doi:10.21105/joss.00861.

[9] Nguyen, Thin, et al. “Prediction of Drug–Target Binding Affinity Using Graph Neural Networks.” 2019, doi:10.1101/684662.

[10] Rogers, David, and Mathew Hahn. “Extended-Connectivity Fingerprints.” *Journal of Chemical Information and Modeling*, vol. 50, no. 5, 2010, pp. 742–754., doi:10.1021/ci100050t.

[11] Tim Dettmers, et al. “Understanding Convolution in Deep Learning.” *Tim Dettmers*, 10 Oct. 2015, timdettmers.com/2015/03/26/convolution-deep-learning/.

[12] Wishart, David S, et al. “DrugBank 5.0: a Major Update to the DrugBank Database for 2018.” *Nucleic Acids Research*, vol. 46, no. D1, 2017, doi:10.1093/nar/gkx1037.

[13] Mikolov, Tomas, et al. "Distributed representations of words and phrases and their compositionality." Advances in neural information processing systems. 2013.

[14] Elif, and Arzucan. “WideDTA: Prediction of Drug-Target Binding Affinity.” ArXiv.org, 4 Feb. 2019, arxiv.org/abs/1902.04166.

[15] Gilson, Michael K., et al. “BindingDB in 2015: A Public Database for Medicinal Chemistry, Computational Chemistry and Systems Pharmacology.” Nucleic Acids Research, vol. 44, no. D1, 2015, doi:10.1093/nar/gkv1072.

# Appendix

Link to GitHub code:
https://github.com/fhooton/DeepDrugRepurposer

As discussed earlier,  Common CNN model always predicted true and hence had a higher F1 score. It can be clearly seen from below confusion metrics.

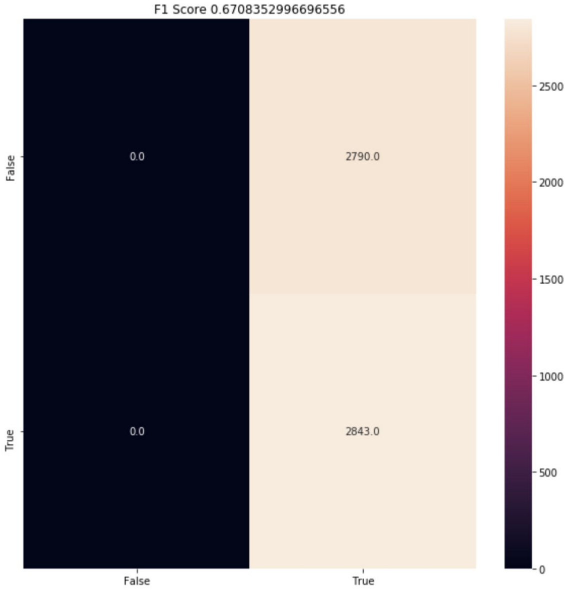


***Figure A1:*** *Shows the confusion matrix for the Common CNN Model's prediction on Test set.*

The promising results were shown by our Deep NN model with an F1 score of 73% and the confusion matrix for the same can be found below:

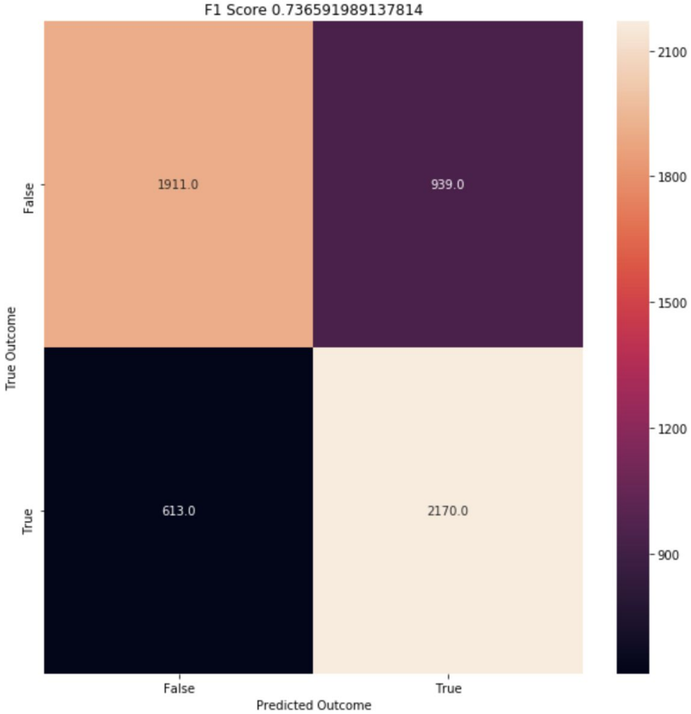


***Figure A2:*** *Shows the confusion matrix for the Common Deep NN Model's prediction on Test set.*

The results imply that Deep NN model is able to draw a decision between positive and negative classifications with some degree of confidence. The CNN's don't perform as well and it may be because of the implications of convoluting word-vectors which are already learned from the dataset.