Predicting the Side Effects of Drugs

Camilo Ruiz*
Department of Computer Science
Stanford University
Palo Alto, CA 94305
caruiz@stanford.edu

Abstract

Predicting the side effects of a given drug is a complicated yet commercially valuable task. Much prior work has focused on utilizing biochemical structural similarity and drug target networks to make such predictions. Here, we utilize word embeddings mined from biomedical corpora for drug targets and side effects to predict the side effects of novel drugs. Ultimately, an LSTM model with attention provides the best performance (AUC = 0.6754) outperforming a logistic regression baseline (AUC = 0.5119).

1 Introduction

1.1 Background and Related Work

The capacity to predict (1) the side effects of new drugs and (2) new but unknown side effects for drugs already on the market is a promising goal. If such a prediction tool existed, pharmaceutical companies would be able to more effectively develop novel drugs with minimal side effects. Simultaneously, warnings could be placed for existing drugs on the market with likely side effects that were not observed in the small clinical trial population.

Historically, approaches to this problem have varied from biochemical structural similarity of distinct drug compounds (Atias et al. 2011) to more recent methods which utilize interactions between the targets of drugs and graph convolutional networks to predict novel drug side effects (Zitnik et al. 2018).

Here, we utilize word embeddings learned from biomedical literature. We learn word embeddings for the targets of drugs as well as side effects and subsequently feed the embeddings to a logistic regression baseline, a single layer neural network, and an LSTM model with attention.

Ultimately, the LSTM model with attention performs the most effectively, garnering an AUC of 0.6754 vs. the logistic regression baseline of 0.5119. Future work will primarily focus on learning new word embeddings for the targets.

2 Problem Statement and Dataset Construction

2.1 Problem Statement

Our ultimate goal is to predict whether a set of drug targets leads to a particular side effect. We conceptualize this as a binary classification problem. For each drug targets-side effect sequence, we output a "1" if the drug targets leads to the side effect and a "0" otherwise.

*Use footnote for providing further information about author (webpage, alternative address)—*not* for acknowledging funding agencies.

Formally: Let d be a particular drug (i.e. Ibrutinib). Drug d functions by targeting a set of set of molecules T_d in a cell (i.e. KRAS, TP53, and BCR). The disruption of the T_d molecules may cause a side effect s (i.e. inflammation) or it may not. To answer whether a given drug d causes a side effect s, therefore, our model functions by: taking as input the set T_d of targets and the particular side effect s; predicting as output a 1 or 0 label corresponding to whether the set T_d led to the side effect s or not.

2.2 Dataset

Our dataset is compiled across three distinct databases, all manually parsed and cleaned.

2.2.1 Drug Side Effects

To extract the side effects of drugs, we utilized the Off-label side effects database (Offsides) (Tatonetti et al. 2012). Offsides contains 487,530 valid drug-side effect pairs between 1,332 drugs and 10,097 side effects. These drug-side effect pairs were filtered based on a significance threshold ($p < 0.001$) to include only drug-side effect pairs of high confidence. Ultimately, this produced 316,913 valid drug-side effect pairs.

2.2.2 Drug Targets

To extract the targets of drugs, we utilized the Search Tool for InTeractions of CHemicals database (STITCH) (Kuhn et al. 2015). For a given drug, STITCH provides a set of targets. For our drugs of interest, stitch provided between 0 and 3000 targets (Figure 1).

2.2.3 Word Embeddings

To ultimately make predictions, we learn word embeddings for each drug (i.e. Ibrutinib), each target (i.e. KRAS), and each side effect (i.e. inflammation). We utilize pre-trained embeddings generated by running word2vec on a biomedical corpus containing: 14 million PubMed abstracts, 700,000 full-text PubMed Central articles, and an English Wikipedia Dump (Moen et al. 2013). Collectively, these corpuses span over 5.5 billion tokens and include all drugs, side effects, and targets in our study. Each word vector has an embedding size of length 200.

2.2.4 Construction of Final Data Set

To structure the data in a manner suitable for classification, we created the following vector for each drug-side effect pair:

- $x = [t_1, t_2, ..., t_n, s]$ where $t_1, ..., t_n$ correspond to the n "highest-confidence" targets (defined below) of a given drug d and s is a side effect. Note that we represent a drug by its targets.
- $y = [1]$ or $[0]$ where 1 corresponds to a valid drug-side effect pair and 0 corresponds to an invalid drug-side effect pair.

Two major decisions were made to compile the final dataset.

First, we set the number n of targets that we would consider for each drug at 5. Note that drugs vary in the number of targets they have. While many drugs have no targets, some drugs can have up to 3500 targets (Figure 1). Since basic neural architectures cannot support variable length inputs and even more sophisticated models like LSTM suffer when input sequences are too long, we had to set up an upper bound on the number of targets we would consider for a given drug. While a larger number n of targets would provide more differentiating information between drugs, it would also reduce the number of drugs we could use. Ultimately, we set $n = 5$ as it allowed to use 294 of the 1332 drugs initially in Offsides (22%).

Second, since Offsides only contains valid drug-side effect pairs, we had to construct invalid drug-side effect pairs. The particular methods used are described extensively in the first experimentation section.

By the end of construction, our data set had:

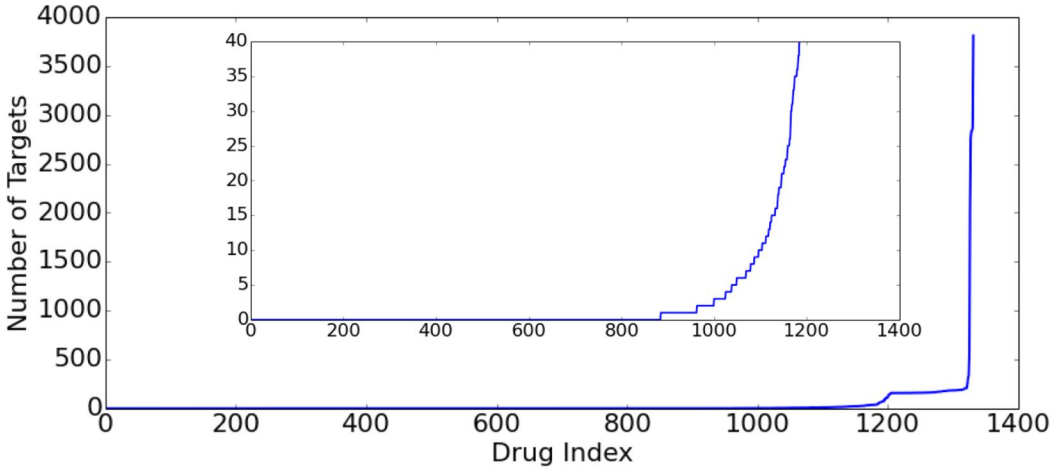

Figure 1: Distribution of Targets by Drug.

- 156,002 valid drug-side effect pairs
- 156,002 invalid drug-side effect pairs

3 Approach

To classify each example, each of our approaches share a common framework (Figure 2):

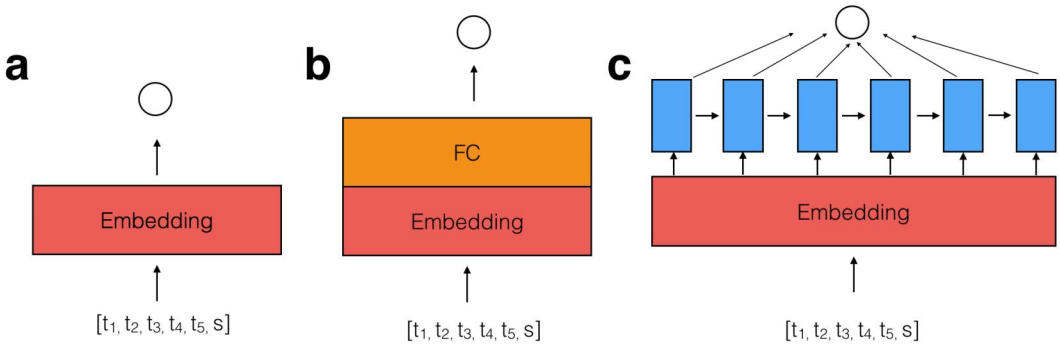

Figure 2: Architectures

1. Map the input sequence $x = [t_1, t_2, ..., t_5, s]$ to its corresponding word embeddings $u = [e_{t1}, e_{t2}, ..., e_{t5}, e_s]$ by indexing into the embeddings matrix E.
2. Run u through either a Logistic Regression, Single Layer Neural Network, or LSTM with Attention Model
3. Use a fully connected layer with a single node and a sigmoid nonlinearity to predict a label 0 or 1 for the input sequence

3.1 Experimental Setup

All models were trained utilizing Keras and TensorFlow on a FloydHub GPU. Models were trained on 60% of the data (187,202 samples), validated on 20% of the data (62,400 samples), and finally tested on 20% of the data (62,400 samples). Across all architectures, we utilized the Adam optimizer and Xavier initialization for weights. When overfitting occurred, we tested all architectures with L2 regularization on either the embedding matrix, the weights in the model, or both. We utilized

Table 1: Performance with Uniform Random Negative Sampling

	Train Acc	Train Loss	Dev Acc	Dev Loss	Test Acc	Test Loss
Logistic Regression	0.6600	0.6166	0.6504	0.6250	0.6409	0.6347
Logistic Regression*	0.8197	0.3881	0.7904	0.5129	0.7918	0.5213
Single Layer NN	0.8094	0.5134	0.7754	0.5896	0.7674	0.5999
LSTM with Attention	0.8149	0.4678	0.7836	0.5447	0.7783	0.5556

$\lambda = 0.0001, 0.001, 0.01, 0.01$ as penalties. Additionally, we tuned the batch size (16384, 8192, 4096) and learning rate (0.0001, 0.001, 0.01) for all models. For the single layer neural network, we additionally tuned the hidden layer size (100, 200, 400). For the LSTM with Attention model, we also tuned the LSTM node size (100, 200, 400). Tuning experiments with graphs are included in supplementary excel files. The architectures with the best performance are reported below.

We first implement a logistic regression model to establish a baseline for classification. We established a baseline both with (*) and without trainable word embeddings.

We next implemented a single layer neural network, hoping that the nonlinearity would improve classification.

Finally, we employed an LSTM with attention model. This approach was motivated by two reasons. First, the sequence of targets is ordered by how strongly the drug interacts with them. Thus, we hoped an LSTM model – by incorporating this information – would improve on the single layer baseline. Second we utilize attention to attempt to "pick" the most relevant target from among the five. This way, the LSTM model could learn to ignore specific noisy targets for each drug when using side effects.

The attention model here consisted of adding a single fully connected layer with a single node sigmoid output. If we consider the sequence $h = [h_{t1}, h_{t2}, h_{t3}, h_{t4}, h_{t5}, h_s]$ to be the outputs of the LSTM, then we calculated $\hat{y} = hW$ where $W \in \mathbb{R}^{a,1}$. W therefore represents an attention matrix, learning the optimal amount to weight each output from the LSTM.

4 Experiments

4.1 Negative Sampling Methods

After parsing Offsides and STITCH, we had a total of 156,002 valid sequences $x = [t_1, t_2, ..., t_5, s]$ with labels $y = [1]$. To generate the negative samples, we followed two approaches.

4.1.1 Balancing Labels across Drugs Only

We began by generating invalid sequences by sampling incorrect side effects uniformly at random.

- For each valid sequence $x = [t_1, t_2, ..., t_5, s_{correct}]$ with label $y = [1]$
- We created an invalid sequence $x = [t_1, t_2, ..., t_5, s_{incorrect}]$ with label $y = [0]$

  $s_{incorrect}$ was drawn from the set of all side effects uniformly at random and verified to not be a valid side effect for the given targets.

This approach generated an equal number of positive and negative examples, such that our data set ultimately had 156,002 valid target-side effect sequences and 156,002 invalid drug-side effect pairs.

Utilizing this version of the data set, we conducted over 20 experiments across the architectures, producing the accuracy and results shown below (Table 1). Ultimately, the logistic regression model with trainable word embeddings showed the best test accuracy (79.04%) and test loss (0.5213). Optimized architectures are shown in Table 2.

Perplexed by the effectiveness of this model, we visualized the weights of the logistic regression unit (Figure 3a). Indices 0:200 correspond to weights for each feature of e_1, the word embedding of target t_1; 200:400 to e_2, 400:600 to e_3, 600:800 to e_4, 800:1000 to e_5, and 1000:2000 to s. Clearly, the

Table 2: Architecture Parameters

	Log Reg	Log Reg, Train Embed	Single Layer NN	LSTM with Attention
Train Embeddings	FALSE	TRUE	TRUE	TRUE
Epochs	150	150	75	50
Batch size	8192	8192	4096	16384
Optimizer	Adam	Adam	Adam	Adam
Learning Rate	0.01	0.01	0.01	0.01
L2 Penalty	0	0	0.0005	0.005
Dense Layer Size			400	
LSTM Unit Size				200

logistic regression model was maximizing its performance by considering the side effect embedding alone.

The fact that learning from just the side effect could produce a model approaching 80% accuracy when the overall dataset was balanced at 50% positive and negative labels implied that a subset of the data was imbalanced. In particular, we hypothesized that the data set might be imbalanced at the level of side effects. Some side effects may be virtually never present in a valid sequence. Conversely, some side effects may be in a valid sequence virtually every time they appear. If this were the case, the logistic regression model could simply predict the most frequent label for every example of a given side effect and achieve good performance.

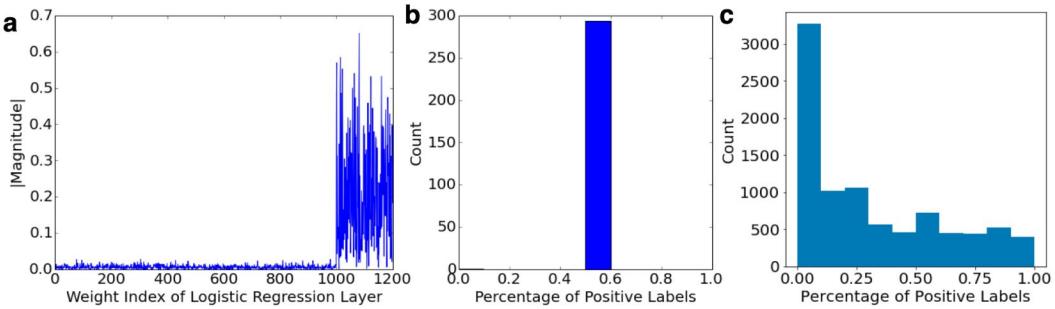

Figure 3: Uniform Random Negative Sampling.

To test this hypothesis, we re-examined the balance of the dataset (Figure 3b,c). While the labels were balanced across drugs (Figure 3b) – i.e. each drug had 50% positive labels and 50% negative labels, the labels were not balanced across side effects – i.e. nearly 3000 side effects virtually never had positive labels (Figure 3c).

The ultimate root of this discrepancy was the uniform random sampling method used to generate invalid examples. In reality, some side effects are extremely rare while others are quite frequent (Figure 4a). By uniformly randomly sampling from the set of possible side effects, we had been introducing many rare side effects as invalid examples with no valid counterparts in the data set to balance them. Ultimately, this generated the class imbalance and prevented our models from learning effectively.

4.1.2 Balancing Labels across Drugs and Side Effects

To improve the balance of the dataset, we decided to sample invalid side effects according to their frequency rather than uniformly. In particular:

- For each valid sequence $x = [t_1, t_2, ..., t_5, s_{correct}]$ with label $y = [1]$
- We created an invalid sequence $x = [t_1, t_2, ..., t_5, s_{incorrect}]$ with label $y = [0]$

  $s_{incorrect}$ was drawn from the set of all side effects according to its frequency rather than uniformly at random (Figure 4a).

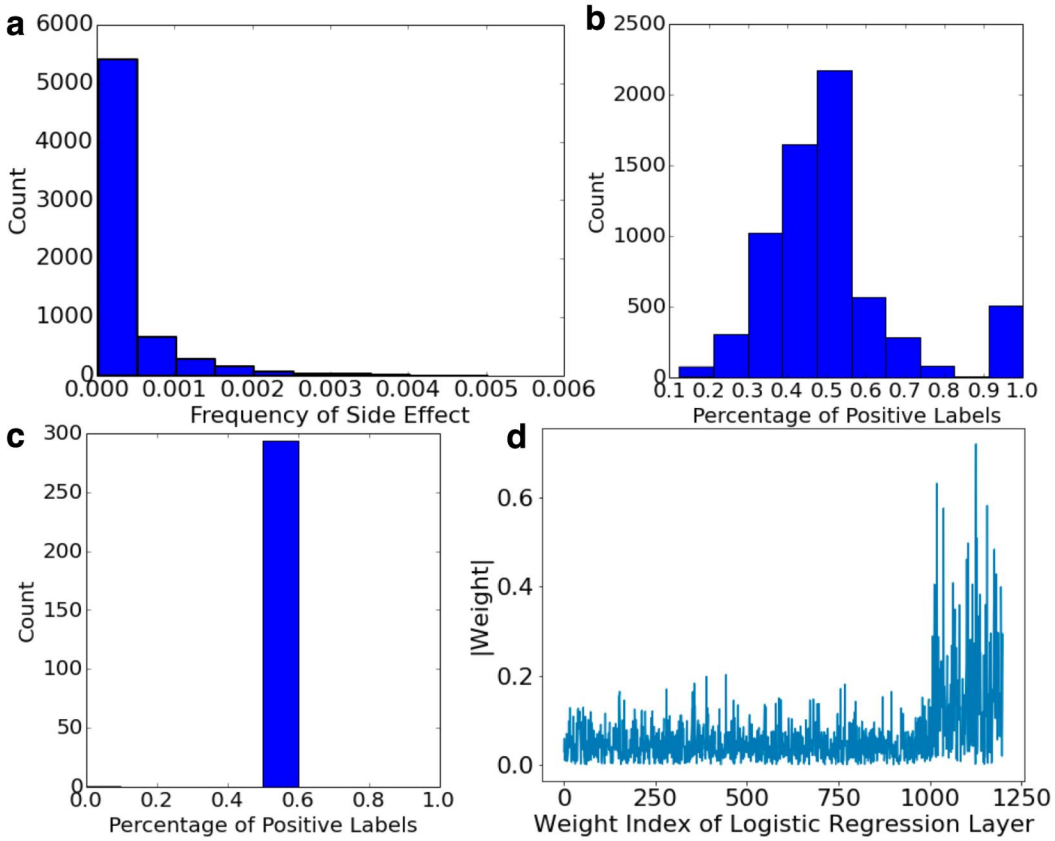

Figure 4: Frequency Weighted Negative Sampling.

Table 3: Performance with Frequency-Weighted Negative Sampling

	Train Acc	Loss	AUC	Dev Acc	Loss	AUC	Test Acc	Loss	AUC
Log. Regression	0.5290	0.6903	0.5429	0.5082	0.6940	0.5119	0.5059	0.6947	0.5119
Log. Regression*	0.5855	0.6606	0.6309	0.4975	0.7770	0.4936	0.4908	0.7816	0.4838
Single Layer NN	0.6462	0.6225	0.7109	0.5872	0.6818	0.6267	0.5911	0.6767	0.6334
LSTM, Attention	0.6174	0.6536	0.6754	0.5887	0.6733	0.6754	0.5951	0.6710	0.6754

By employing this approach, we were able to more effectively balance the dataset across both side effects and drugs. The data set is perfectly balanced across all drugs (50% valid examples and 50 % invalid examples) (Figure 4c). Moreover, the dataset is relatively balanced across side effects (Figure 4b). Most side effects have roughly 50% positive labels, although a few exceptions exist. Finally, to verify that the logistic regression model was indeed learning from the target embeddings, we plotted the weights of the logistic regression layer (Figure 4d). Compared to the weights for the unbalanced dataset (Figure 3a), the weights corresponding to the target embeddings are substantially higher. The model still weighs the side effect much more heavily than the target embeddings, but that may simply suggest that the side effect embedding now carries more differentiating information.

4.2 LSTM Attention Analysis

With a balanced dataset, we reran all experiments and tuned the corresponding models. Ultimately, the LSTM with attention model showed the best performance (Table 3, Table 4).

Table 4: Architecture Parameters

	Log Reg	Log Reg, Train Embed	Single Layer NN	LSTM with Attention
Train Embeddings	FALSE	FALSE	FALSE	FALSE
Epochs	50	50	150	80
Batch size	16384	16384	16384	16384
Optimizer	Adam	Adam	Adam	Adam
Learning Rate	0.01	0.01	0.01	0.01
L2 Penalty	0	0	0	0.001
Dense Layer Size			200	
LSTM Size				200

4.3 Error Analysis

Manually inspecting samples proved challenging as no one is currently capable of looking a set of protein targets and knowing whether it will cause a side effect. We therefore asked three directed questions to understand when our model was making errors.

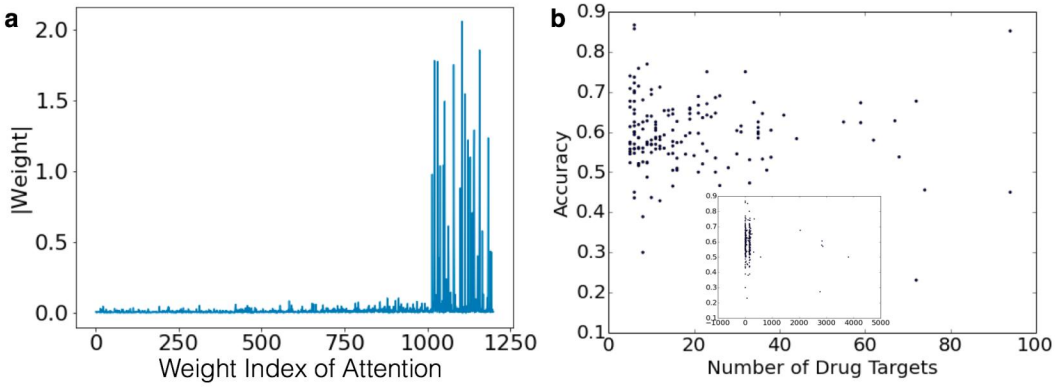

Figure 5: LSTM with Attention Error Analysis.

4.3.1 Is the Attention heavily prioritizing the side effect information?

First, we visualized the weights of the attention layer of the LSTM model (Figure 5a). The first 1000 indices correspond to the hidden layer outputs of the LSTM for the drug targets; the last 200 indices correspond to the hidden layer outputs for the side effect. Again, the model heavily prioritizes the side effect, implying that relatively little information is captured in the embeddings of the targets.

4.3.2 Should we increase the number of drug targets considered?

We further suspected that our initial choice of using just five drug targets may be limiting the performance of our model. Indeed, five targets may capture the information content of drugs with ten targets well but may not capture the information of drugs with hundreds of targets as effectively. Similarly providing more targets may provide additional differentiating information, thus potentially causing attention to consider the targets more with respect to the side effect.

To test whether increasing the number of drug targets was likely to improve accuracy, we ran the LSTM with attention model using sequences of variable length ranging from 1 (side effect only) to 5 (side effect plus four targets) (Table 3). Ultimately, adding the first target leads to a substantial decrease in loss and boost in accuracy. Adding additional targets also decreases loss and increases accuracy but with marginal returns.

Table 5: Performance vs. Number of Targets Considered

	Train Acc	Loss	AUC	Dev Acc	Loss	AUC	Test Acc	Loss	AUC
5 Targets + SE	0.6174	0.6536	0.6754	0.5887	0.6733	0.6325	0.5951	0.6710	0.6379
4 Targets + SE	0.6237	0.6559	0.6793	0.5929	0.6775	0.6332	0.5976	0.6737	0.6403
3 Targets + SE	0.6181	0.6587	0.6701	0.5911	0.6780	0.6282	0.5942	0.6748	0.6346
2 Targets + SE	0.6114	0.6638	0.6610	0.5844	0.6801	0.6214	0.5890	0.6771	0.6276
1 Target + SE	0.5861	0.6745	0.6277	0.5723	0.6831	0.6019	0.5702	0.6824	0.6023
SE	0.5246	0.6915	0.5374	0.5177	0.6921	0.5273	0.5137	0.6931	0.5202

4.3.3 Accuracy across Drugs vs. Number of Drug Targets

Finally, we compared our accuracy across drugs based on the number of drug targets (Figure 5b). There was very little correlation between accuracy and the number of drug targets.

Ultimately the analyses above suggested that the primary limitation of the LSTM with attention model had less to do with the architecture of the model than with the word embeddings themselves. Indeed, the embeddings learned for the targets may not carry sufficient meaningful information, thereby limiting the model. Adding more targets was unlikely to help.

4.3.4 The Potential of Word Embeddings

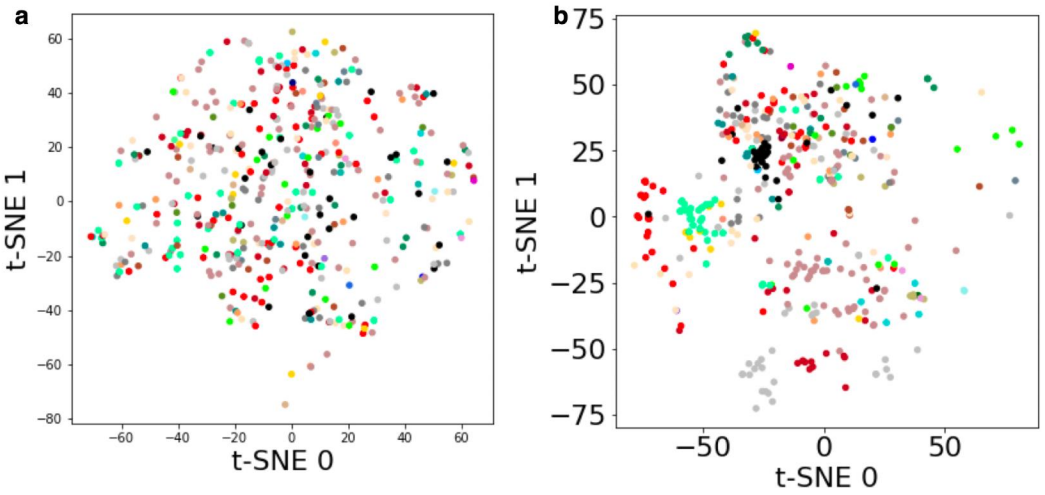

Figure 6: Word Embeddings for Side Effects.

As a final test for whether the word embeddings carried meaningful information, we conducted t-SNE to visualize the embeddings corresponding to the side effects. We categorized the over 2000 side effects into 32 categories based upon the anatomical region of the body they affected (i.e. gastrointestinal system) (Zitnik et al. 2018). Each point on the t-SNE plot represents a side effect; it's color represents the anatomical system it affects. We compared the t-SNE of the pre-trained word embeddings (Figure 6a) to those learned by the best LSTM with Attention Model that used trainable word embeddings (Figure 6b). Note that ultimately the best models in terms of AUC did not have trainable word embeddings.

We can see that the pre-trained word embeddings carry little structural information regarding this classification. By contrast, the word embeddings learned from the LSTM with attention model carry some rough clustering patterns. Red, for example, corresponds to the gastrointestinal system and clusters roughly whereas it did not before.

We could not produce the same t-SNE plot for the drug targets as there would have been no analogous way to interpret it. However, it seems like the models do have the potential to learn appropriate

word embeddings for the side effects, even if not the targets. As a result, future work utilizing the word embeddings for the side effects and an orthogonal embedding system for the targets could prove promising.

5 Future Work

In order to improve the model, we suggest three major changes. First and most importantly, we suggest the use of a distinct corpus to train the word embeddings for the targets. We suspect that by including PubMed Abstracts, PMC Articles, and Wikipedia and learning word embeddings in an unsupervised way, the ultimate word embeddings were not particularly meaningful. We suspect that removing Wikipedia could improve the signal contained within the embeddings. When drugs and side effects are mentioned in the biomedical literature, they are likely mentioned with highly specific contexts that are differentiating. By contrast, when drugs and side effects are mentioned in Wikipedia, the context is likely less technical and more similar across different drugs and side effects. As a result, the overall embedding is less meaningful. Similarly, the usefulness of the word embeddings could be improved by testing different embedding sizes.

Second, additional external data could be utilized to augment the information attributed to each target. Targets existing in a graph of interaction nodes (a protein-protein interaction network). By utilizing a method such as node2vec to learn a graph-based embedding for each node, additional information would be carried by each target. The graph and word embeddings could subsequently be jointly fed into the attention model to analyze the relatively utility of word embeddings vs. graph embeddings.

Finally, more complex attention models could be utilized to improve predictions. For example, the output of the hidden layer corresponding to the side effect could be utilized as a query vector over the outputs of the hidden layers corresponding to the targets. The attention would then be more sophisticated, not just learning which of the LSTM outputs is most meaningful (our model) but also being able to modify which LSTM unit is considered for each new sample.

Acknowledgments

We would like to thank Professor Richard Socher for mentoring this project. Similarly, we would like to thank Dr. Marinka Zitnik for providing helpful conversations regarding next steps.

References

[1] Atias, N., Sharan, R., 2011. An algorithmic framework for predicting side effects of drugs. J. Comput. Biol. 18 (3), 207218. [2] Hochreiter, Sepp, and Jrgen Schmidhuber. "Long short-term memory." Neural computation 9.8 (1997): 1735-1780. [3] Moen, S. P. F. G. H., and Tapio Salakoski2 Sophia Ananiadou. "Distributional semantics resources for biomedical text processing." Proceedings of the 5th International Symposium on Languages in Biology and Medicine, Tokyo, Japan. 2013. [4] Kuhn, Michael, et al. "STITCH 3: zooming in on proteinchemical interactions." Nucleic acids research 40.D1 (2011): D876-D880. [5] Nicholas P. Tatonetti, Patrick P. Ye, Roxana Daneshjou, and Russ B. Altman. "Data-Driven Prediction of Drug Effects and Interactions" Sci Transl Med 14 March 2012 4:125ra31. [DOI:10.1126/scitranslmed.3003377] [6] Wang, Yequan, Minlie Huang, and Li Zhao. "Attention-based lstm for aspect-level sentiment classification." Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing. 2016. [7] Zitnik, Marinka, Monica Agrawal, and Jure Leskovec. "Modeling polypharmacy side effects with graph convolutional networks." arXiv preprint arXiv:1802.00543 (2018).