Predicting Drug-Drug Interactions

Team members: Vikram Reddy, Bhuvana Bellala, Sameer Bajaj, Nicolas Mon

Introduction:

Drug to drug interactions are a subcategory of adverse drug reactions (ADRs). According to a study in 2013, ADRs account for 300,000 deaths annually across Europe and the United States (Businaro, 2013). The latest biomedical research papers describe new drugs that come on the market and their potential adverse effects. The automatic extraction and classification of these potential drug to drug interactions would assist a doctor in deciding whether or not to prescribe a certain new drug to a patient.

Our project proposed to classify drug pairs as exhibiting an interaction or not. We used the dataset from the 2013 DDI Extraction Challenge. It consisted of 792 biomedical texts from DrugBank. These texts were parsed into sentences and then run through MetaMap to extract potential drug pairs. MetaMap was used to identify sentences that contain pharmacological substances as detected by Metathesaurus. Each sentence was then manually annotated by experts to create a dataset of statements that unambiguously identify 1) interacting drug pairs, and 2) non-interacting drug pairs[1] (Aronson). The Inter-Annotator Agreement for the DrugBank documents were all above a Kappa coefficient of 0.80, with some reaching a Kappa of 0.96. For the Medline abstracts, the Inter-Annotator Agreement was between 0.55 to 0.72 (Herrero-Zazo et. al, 2013).

Our goal was to featurize each sentence in the parsed biomedical data, associate that sentence or fragment to a drug pair, and train a classifier to predict future drug pairs. The state-of-the-art F1 score during the challenge was 67, and we had an F1 score of 59 on our best result. Our best model used lexical, syntactic, and semantic features and ran a cross-validated random forest as the classifier.

Data Challenges:

There were several challenges in the data that we addressed during preprocessing and featurization. First, in a single sentence there might be more than 2 drug names. If so, each sentence would have 3 or more drug pairs associated with it. To differentiate the data associated with each drug pair, we split up the sentence into 3 fragments. These fragments--before drug 1, between drug 1 and drug 2, and after drug2--were treated as

separate bag of words for each drug pair. In this way, the feature vectorization for each drug pair would be slightly different, which might mitigate the effects of drug name repetition.

Secondly, a preliminary analysis showed the dataset is highly unbalanced towards non-interacting drug pairs. We took two measures to account for this skew. Based on the suggestions from Bobic, Fluck and Hofmann-Apitius, we performed random undersampling of the data and obtained a balanced dataset with about 50% interacting pairs and 50% non-interacting pairs[11]. Second, we primarily used the F1 score to judge our results. Since F1 score is a measure of both precision and recall, we were able to drive the feature set development based on how well our classifier is performing on the interacting drug pairs.

Thirdly, if the name of a drug, say 'drug1', appeared more than once in a sentence, the corpus included the pair <'drug1', 'drug1'> as a potential interacting pair. However, after reading Bui's paper, we excluded such pairs to avoid confusing our model[2].

Finally, we augmented the dataset by adding three key columns. The first two columns are called 'drug1offset' and 'drug2offset'. These two columns represent the character offsets of the two drugs in a given sentence. For example, in the sentence 'Population pharmacokinetic analyses revealed that MTX, NSAIDs, corticosteroids, and TNF blocking agents did not influence abatacept clearance', MTX is at 50-52 and TNF blocking agents is from 84-102. These two columns helped us segment the sentences correctly during feature extraction because some drugs were spread across the sentence or appeared more than once within the same sentence. The third column is called 'drugsinsent'. This column contained all the drugs that were mentioned in a given sentence. Using this column, we were able to quickly build features that took into account the relation between all the drugs mentioned in a sentence.

Two-Stage Classification:

Another way to address the unbalanced data was to apply a 2-stage classification model. This is used if we were to classify the interacting pairs further into 4 subcategories--int, advice, mechanism, and effect. One of the teams in the 2013 DDI Extraction Challenge, UWM-Triads, used this in combination with preprocessing to deal with the unbalanced data. About four-fifths of the data exhibited non-interactions and within the interactions, only about 1/15 were characterized as "int". Thus, the training set was heavily skewed.

As researcher Majid Rastegar-Mojarad explains, data should be preprocessed to prepare for the first stage of binary classification, then post-processed to prepare for the second stage of multivariable classification (Rastegar-Mojarad). We applied this methodology to the baseline model using only TFIDF features. It was beyond the scope of our project to classify interactions into 4 categories, so we left the 2-stage classification out of the final pipeline. We have included it in the code as a separate file. We will use this method for future classification work with unbalanced data.

Feature Extraction/Description of Algorithms:
We used many feature extraction methods that we read about in various research papers. We read about 15 papers from the various participants in the data challenge, as well as from other DDI Extraction researchers. We have listed these papers in the Works Cited section.

Our features used a bag of words model, regex parsing, and dependency parsing. For all the features in the bag of words models, we first segmented the sentence into three parts. The first segment contains the words before the mention of the first drug, the second segment contains the words in between the two drug mentions, and the last segment contains the words after the second drug mention. After segmenting the sentence into three parts, we built the following features:

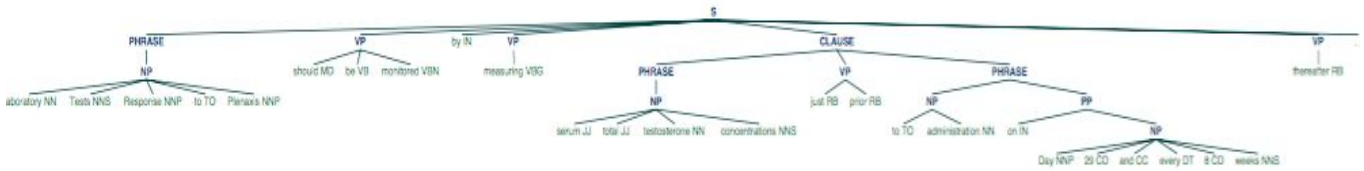
Lexical Features:
In order to capture the token information around each pair, for each sentence fragment, we counted the number of words, the number of drug mentions, the number of verbs in a parse tree, the number of trigger words, and the number of negation words. We researched a list of trigger words, such as “administer”, “concomitantly”, “increase”, “decrease”, and “absorption”, and inferred that these words would be good indicators of an interaction (Bui). We also researched a list of negation words, such as “not”, “neither”, and “unable” (Chowdhury 2011) to account for negation of a trigger, such as “does not increase” or “unchanged”.

In addition, we also implemented a feature that returned the distance of the closest trigger word to a drug mention. The idea behind this feature was to indicate sentences in which a trigger word appeared very close to a drug mention, as trigger words closer to drug mentions could mean a higher chance that that sentence indicates a reaction. Another reason to calculate the distance from trigger to negation and from negation to the trigger

is to incorporate relationships between words in the sentences. Some sentences have repetitions of the same words (as described in the Data Challenges section) that appear in different parts of the sentences. One method to address this would be to simply delete the repetitions during normalization, but we decided that this would be a loss of information. Instead we tried to encode this distance in the features. Granted, the distance count is a loose feature for this end goal. After this idea, we were motivated to implement dependency parser and word2vec features to get a better representation of the grammatical and semantic relationships within the text.

Dependency parser features:

Many of the sentences in our dataset have a very complex structure, and some words were related in ways that were not apparent in simply the location of the word in the sentence. In order to first understand the syntactic nature of our sentences, we built a parser that chunked our sentences, into noun-phrases, verb-phrases, prepositional-phrases, phrases (contained one or more noun- and prepositional-phrases), and clauses (a phrase followed by a verb phrase followed by a noun phrase). The parser was modeled after the suggestion made by Bui and et al. Here is a sample of the tree built by our parser[2]:



After analyzing the syntactic structure of several sentences, we decided to use a dependency parser to capture these relationships. Using the spacy dependency parser, we built the following features:

- One feature attempted to capture the 'subject-direct object' relationship that would be present in sentences like 'Drug1 effects Drug2'. In that sentence, Drug1 is the subject and Drug2 is the direct object. Oftentimes, however, this relationship would not be as straightforward. For example, you could have the sentence 'Drug1 increases the absorption of Drug2'. In this case, absorption is the direct object and Drug2 is in a subordinate prepositional phrase. To address this, the

'subject-object' feature searched the entire subtrees of the subject and direct object of the sentence, returning 1 only if one drug was the subject or in its subtree, and the other was the direct object or in its subtree.

- Another feature implemented using the dependency parser searched for a trigger word in the dependency path from each drug mention to the root verb of the sentence. A third binary feature using the Spacy dependency parser would check if both drug mentions had the same head, meaning they both depended on the same word.

Sentiment Analysis:

Based on the intuition that interactions between two drugs could either be positive or negative, we investigated the effect of sentiment analysis features in locating drug interaction mentions in sentences. We addressed the task of locating drug interactions mentions in the sentence text using heuristics to measure the strength of positive or negative sentiment in the sentence text, so that we can classify the remaining neutral sentences as non-interactions. Given the input text, we defined an algorithm that would calculate the overall sentiment value to provide a positive, negative, neutral and compound score to the sentence. With sentiment analysis, we saw marginal improvements in accuracy and F1 score. It was not surprising to us to see a small change in accuracy due to the complex sentence structure of the dataset. For example, the sentence text - 'Although MIVACRON (a mixture of three stereoisomers) has been administered safely following succinylcholine-facilitated tracheal intubation, the interaction between MIVACRON and succinylcholine has not been systematically studied,' scored a high negative score and a mid-range neutral score on our sentiment analysis algorithm, however, the drugs mentioned in the sentence do not interact with one another. With the results of the F1 Score, we realized that sentiment analysis would only have a marginal effect on improving the accuracy of our model.

Word2Vec Features:

Another helpful feature to learn the semantic meaning of the words is to use word embeddings. The word2vec module of gensim groups contextually similar words together. The vectorization and dimensionality reduction of these words would help to extract the concepts from the text. The bag of words approach, by contrast, is prone to problems of homophony and polysemy (Crain). To take advantage of the semantic meaning around each drug entity, we ran the gensim model on the sentences in out train

dataset. Then for each sentence within our dataset we tokenized and tagged the sentences. In order to capture the immediate context of each drug and the relationship between the two, we traversed through the sentence and found the closest verb or noun and vectorized the token using our word2vec model. Once vectorized, we took the mean of the model. Our intention with this feature was to build a classifier that understood the context of a given entity pair. With the addition of this feature, we saw a marked increase in out F1 score from 58 to 59.1.

Classifiers and GridSearchCV:

Initially, we used a MultionomialNB with alpha=1.15 as our classifier. However, even with the addition of multiple features, we never saw an improvement in out F1 Score from 42. So, we tried Logistic regression, SVM, AdaBoostClassifer, KNeighborsClassifier, and RandomForestClassifer. For each classifier, we ran a GridSearchCV in order to fine tune the parameters. In the end, our best classifier was the RandomForestClassifier with an accuracy of 80.53% and an F1 score of 59.75 for the interacting drug-drug pairs. For the RandomForestClassifer, the GridSearchCV optimized the n_estimators and max_depth parameters.

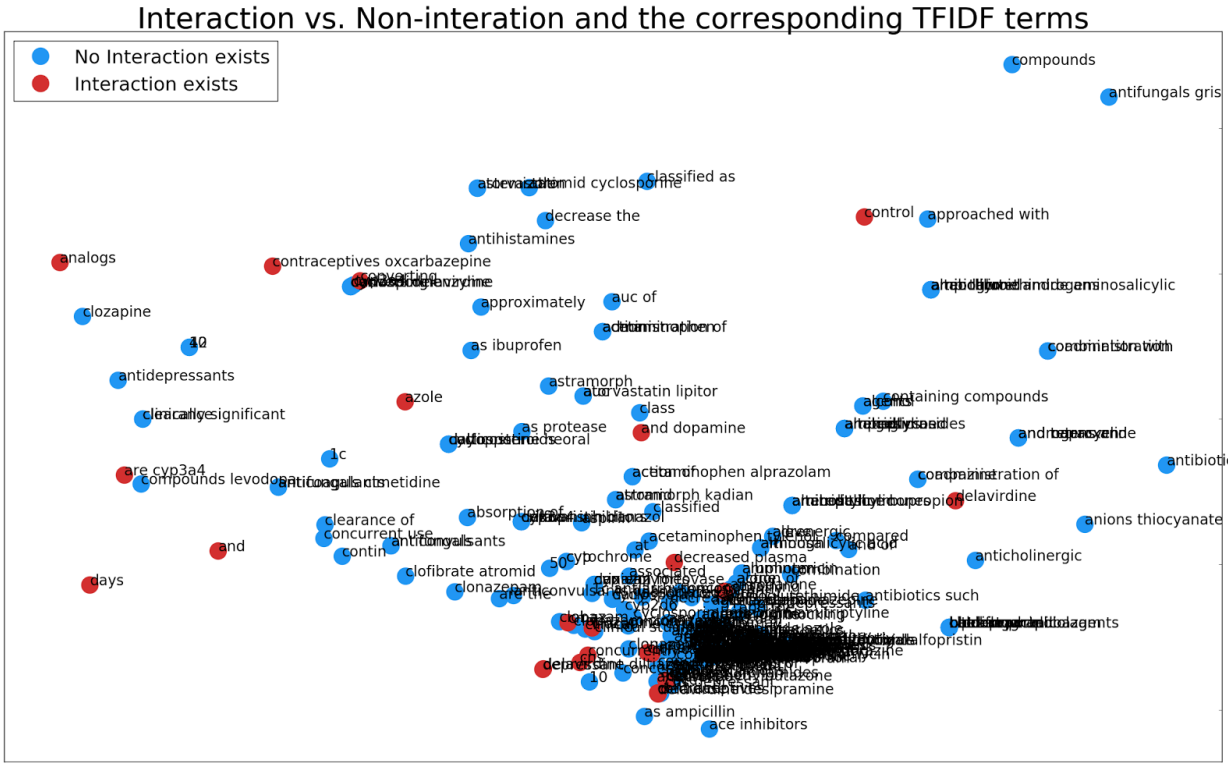
Additional Feature Selection:

One way to computationally reduce the size of our feature set was to use the Variance Threshold functionality. We set a minimum for the within-feature variance when running without the dependency parser and word2vec features. This improved the accuracy from .28 to .39, so we reasoned that many of the simple bag of words features did not contribute as much to the score.

Error Analysis:

After extracting several features and generating the predicted output, we created an excel spreadsheet for error analysis. Each row had the sentences, the drug pairs, the expected value, the predicted value, and the list of contributing features. From this document, we were able to pinpoint for which sentences our algorithm made mistakes. We also found new challenges in the data. For example, one sentence had an ambiguous description--that one drug would increase, decrease, or have no effect on the second drug. This drug-drug pair was manually annotated as having an interaction, despite its ambiguity. We realized that drugs having a small fractional chance of an interaction, should be classified as a 1, so our algorithm had to function accordingly.

Also, to pinpoint what terms resulted in interactions and non-interactions, we visualized the data on the training set. We first chose a random sample of 300-600 terms and then used multidimensional scaling to map the features to a 2 dimensional space. Then we used matplotlib with d3 tooltips to identify which key terms resulted in positive interactions. The code for this was largely used from Brandon Rose's clustering notebook (Rose). These key terms were used to infer possible feature extraction. There were several interactions with types of enzymes labeled. For example, the bigram "are cyp3a4" is a reference to an enzyme which is involved in digestion. We reasoned that it could be classified as an interaction because the biomedical research articles that contain this term would be discussing the drug's interaction with this particular enzyme. Thus, for future work, we could collate a list of known enzymes in drug to drug interactions, which we could append to our list of key "trigger" words.



Results:

Our baseline accuracy established by running a MultinomialNB on a vectorized bag of words was 42.965% on the pseudo-balanced dataset. With the addition of features and model optimization, we were able to increase the accuracy up to 80.532%.

Also, our baseline F1 score for interacting drug pairs was 0.36. We have come a long way by customizing our features and optimizing our models to reach an F1 score of ~0.597. A number of useful features such as locating trigger words, drug names and running dependency parsers helped increase the F1 score of our model. Apart from experimenting with numerous features, we also tried our best to find the most optimal model for our dataset. By tuning parameters, and running a grid search, we were able to achieve a considerably high F1 score of 0.6.

Future work:

Since the 2013 DDI Challenge, there has been new research using the same dataset. For example, Zhao uses convolutional neural networks to predict interactions and achieves an F1 score of 68.8, which beat the winner of the challenge (Zhao et al., 2016). Zhao's CNNs were based on syntactic word embeddings. If we were to continue our project, we would use our word2vec features as inputs to a nonlinear neural network. In the current state of our project, we used word2vec features as inputs to a linear model. We would also try using a semantic parser to extract the relationships between the words rather than just having collocations or keywords as features in isolation.

In addition, another way we could expand on our project would be to change our classifier from a binary classifier (predicting interactions or non-interactions) to one that predicted what type of interaction that the sentence signified. This would use a 2-stage classification as described earlier.

Contributions of each team member:

We split the work into three distinct stages data preprocessing, feature extraction, and model optimization. Most of the features were generated together.

Word	**Team Member**
Preprocessing - Vikram: Extracting numbers from the dataset to the 'NUM' tag - Bhuvana and Vikram: Extracting the sentences from XML file into sentence, ddi_label, ddi_pair, entity_id	Vikram and Bhuvana

- Bhuvana: Extracting character offsets as well from the XML file	
Feature Extraction - Dependency parsing feature prototyping: all - Dependency parser implementations: Nic - Lexical features: Nic, Vikram, Sameer, and Bhuvana - Word2Vec features: Vikram and Bhuvana - Sentiment Analysis features: Sameer and Bhuvana	Vikram, Bhuvana, Sameer, and Nic
Model optimization - Feature selection with Variance as a parameter: Nic and Vikram - GridsearchCV: Nic, Sameer, Bhuvana, Vikram	Vikram, Bhuvana, Sameer, and Nic
Poster and Final write-up	Vikram, Bhuvana, Sameer, and Nic
Error Analysis - Comparison of Predicted vs. Actual values: Sameer - Visualization of the training set: Vikram	

Works Cited:

[1] Aronson, Alan R. Effective Mapping of Biomedical Text to the UMLS Metathesaurus: The MetaMap Program. 2001. National Library of Medicine. Bethesa, MD 20894.

[2] Bui, Quoc-Chinh, Peter M.A. Sloot, Erik M. van Mulligen, and Jan A. Kors. A novel feature-based approach to extract drug–drug interactions from biomedical text Bioinformatics

(2014) 30 (23): 3365-3371 first published online August 20, 2014 doi:10.1093/bioinformatics/btu557

[3]Chowdhury, Md. Faisal Mahbub et. al. Exploiting the Scope of Negations and Heterogeneous Features for Relation Extraction: A Case Study for Drug-Drug Interaction Extraction. 2011.

[4]Crain, Steven P. et al. Dimensionality Reduction and Topic Modeling: From Latent Semantic Indexing to Latent Dirichlet Allocation and Beyond. DOI: 10.1007/978-1-4614-3223-4_5. In book: Mining Text Data, pp.129-161

[5]He L, Yang Z, Zhao Z, Lin H, Li Y (2013) Extracting Drug-Drug Interaction from the Biomedical Literature Using a Stacked Generalization-Based Approach. PLOS ONE 8(6): e65814. doi: 10.1371/journal.pone.0065814

[6]María Herrero-Zazo, Isabel Segura-Bedmar, Paloma Martínez, Thierry Declerck, The DDI corpus: An annotated corpus with pharmacological substances and drug–drug interactions, Journal of Biomedical Informatics, Volume 46, Issue 5, October 2013, Pages 914-920, http://dx.doi.org/10.1016/j.jbi.2013.07.011.

[7]Miwa M, Rune Saetre, Yusuke Miyao, Jun-ichi Tsujii. Proceedings of the 2009 Conference on Empirical Methods in NLP. ACL. Singapore; 2009. A rich feature vector for protein-protein interaction extraction from multiple corpora; p. 121-130.

[8]Isabel Segura-Bedmar, Paloma Martínez, María Herrero Zazo, (2014). Lessons learnt from the DDIExtraction-2013 shared task, Journal of Biomedical Informatics, Vol.51, pp:152-164.

[9]María Herrero-Zazo, Isabel Segura-Bedmar, Paloma Martínez, (2013). Annotation Guidelines. http://labda.inf.uc3m.es/DrugDDI/annotation_guidelines_ddi_corpus.pdf

[10]Rose, Brandon. Document Clustering with Python. http://brandonrose.org/clustering

[11]Bobic, Tamara et. al. SCAI: Extracting drug-drug interactions using a rich feature vector. 2013. Atlanta, Georgia. Association for Computational Linguistics

[12]Thomas, Phillippe et. al. WBI-DDI: Drug-Drug Interaction Extraction using Majority Voting. 2013. Atlanta, Georgia. Association for Computational Linguistics

[13]Chowdhury, Md. Faisal Mahbub. FBK-irst : A Multi-Phase Kernel Based Approach for Drug-Drug Interaction Detection and Classification that Exploits Linguistic Information 2013. Atlanta, Georgia. Association for Computational Linguistics

[14]Bobic, Tamara et. al. SCAI: Extracting drug-drug interactions using a rich feature vector. 2013. Atlanta, Georgia. Association for Computational Linguistics

[15]Rastegar-Mojarad, Majid, Boyce, Richard D, Prasad, Rashmi (2013). UWM-TRIADS: Classifying Drug-Drug Interactions with Two-Stage SVM and Post-Processing. http://www.aclweb.org/anthology/S13-2110

[16]Hailu, Negacy D, Hunter, Lawrence E, Cohen, K. Bretonnel (2013). UColorado SOM: Extraction of Drug-Drug Interactions from BioMedical Text using Knowledge-rich and Knowledge-poor Features. http://www.aclweb.org/anthology/S13-2112

[17] Zhao Z, et. al. Drug drug interaction extraction from biomedical literature using syntax convolutional neural network. 2016. DOI: 10.1093/bioinformatics/btw486