Discovering Adverse Drug Reactions via Natural Language Processing of Twitter Posts

Benjamin Pastel
bpastel@stanford.edu

Blanca Villanueva
villanue@stanford.edu

Abstract

Several studies show that user-posted data on social media sites can be used to detect adverse drug reactions (ADRs) faster than typical methods (e.g., via reporting to the FDA Adverse Event Reporting System), and throughout a drug's market lifetime. Previous studies have sourced data from search queries and social media posts and have tested several NLP and machine learning methods to yield increasingly accurate results. This study aims to emulate published results by optimizing for effective input vectors and using more advanced deep learning models to classify ADRs present in Twitter posts. Our baseline model (logistic regression) was able to achieve an F1 score of 0.43 (versus the state-of-the-art ADR Class F1 of approximately 0.57 on a comparable data set) using word-presence vectors created using the GloVe Twitter vocabulary and a 3-layer Feed Forward Neural Network.

1 Background

Adverse Drug Reactions (ADRs) are harmful effects that are caused by the non-abusive consumption of medication. After a drug passes clinical trials and is released to the market, it is still important for health professionals to be able to detect, assess, understand, and prevent ADRs related to said drug. This set of activities is commonly referred to as pharmacovigillance.

A number of studies have used NLP techniques on publicly available data to discover ADRs. Several of these studies use data from online fora specifically related to medical experiences (e.g., Medline, AskAPatient); few make use of more general social media sites, and few use advanced deep learning techniques for classification.

This is a challenging NLP task for several reasons: the format of Twitter posts is highly informal, and a minority of these posts include personal medical information; negative sentiment does not directly translate into an ADR mention (i.e., the task is not simple sentiment classification). In addition to applying more advanced techniques to this domain, this study is also an exploration of the feasibility of non-medical social media as data sources for pharmacovigillance efforts.

2 Data

The data were collected from a study conducted by the Diego Lab at Arizona State University[2]. The data set consists of 7,281 Twitter posts manually labelled on whether or not the tweet contains an ADR mention. The tweets were annotated by two domain experts and a pharmacology expert. The tweets were pre-selected to contain a drug mention. The ratio of tweets with ADR mentions to those without is roughly 9:1.

For example, here are 3 randomly selected tweets with ADR:

```
@GangamStyleDad is that like quetiapine. when i had to take that crap
i could not stay up longer than 11 hours.
```

```
I reaaaallly need to take my Paxil but it makes me feel so delirious
and just messed up.

This quetiapine isn't working damnit!!!! Does make me hungry
at night though... *sigh*
```

and here are 3 randomly selected tweets (after filtering for obscenity) with a drug mention but no ADR:

```
@suedovecote I don't think it does,not when I take it anyway.
#fluoxetine

@HockeyBroad halls. Totally losing my voice and need a lozenge.
Haha.

Xarelto side effect: "may cause bleeding, most of which is serious
and sometimes leads to death" That sounds fantastic.
```

2.1 Word Embeddings

While Sarker et. al. create word embeddings from the corpus of collected tweets, we use several different vector representations for inputs:

1. Word Presence vectors, similar to the BOW representation, except that it does not track word frequency within the tweet
2. Average of word vectors present in each tweet, based on the GloVe Twitter word vectors (Pennington, Socher, and Manning 2014)
3. For the RNN and CNN models, we used left-to-right sequences of GloVe word vecs; either truncated to the first n words, or windows centred on the first drug mention

2.2 Additional processing of GloVe Twitter vectors

For the first implementation using pre-trained GloVe word vectors, we do not directly input GloVe word embeddings into our model. The word vectors for each word in a tweet are averaged together such that each tweet is represented by a single input vector. For word vectors not included in the GloVe Twitter vocabulary, we simply use the zero vector, such that the resulting input vector effectively ignores any words missing from the GloVe Twitter vocabulary.

2.3 Train, Development, Test Sets

For our baseline model, we split the 7,281 tweets into stratified train, development, and test sets.

Table 1: Train, Development, Test Split of the Data

	# Tweets
Train	4,281
Dev	1,000
Test	2,000

3 Models

3.1 Logistic Regression

Logistic regression is well suited for this binary classification task because it can handle large feature spaces. We compare two different input types: (1) word-presence vectors of dimension the vocabulary, with a 1 in each index if the corresponding word is present in the tweet, (2) the average of GloVe word vectors per tweet.

3.2 Feed Forward (FF)

Our 3-layer feed forward network contains an input layer, 1 hidden ReLU layer, and a tanh activation layer; dropout occurs at all layers. We compare the same two input types as with the LogReg model. Optimized hyperparameters include: dropout for each layer, learning rate, dimensions of the word embeddings, number of hidden units, and the weight attached to the positive (ADR) class. Published research (Iyyer et al. 2015) indicates that simpler models that do not incorporate word order can outperform more complex methods that do, even in cases where input sentences are short (e.g., tweets).

3.3 Recurrent Neural Net (RNN)

Because tweets are relatively short, we experiment with a simple RNN model to see whether or not the model weights are able to capture enough information about tweet semantics. Though tweet length typically reaches 20 words, we explore the hyperparameter space to address the performance limitations of RNNs related to input length. Our RNN compares two different input types: (1) concatenated GloVe word vectors for a fixed window beginning from the start of the tweet, (2) concatenated GloVe word vectors centred on the drug mention. Optimized hyperparameters include: dropout for the hidden layers, learning rate, dimensions of the word embeddings, L2 regularisation, and the weight attached to the positive (ADR) class.

3.4 Convolutional Neural Net (CNN)

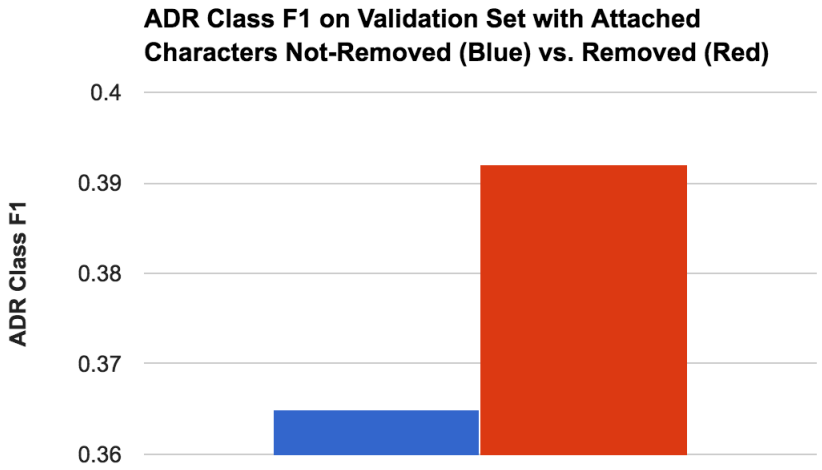
We also implement a CNN model to explore whether or not this model can capture any finer-grained semantics within the n-grams in the inputs. Our CNN compares the same two input types as the RNN model. Optimized hyperparameters include: dropout for the max-pool layer, learning rate, max sentence length, L2 regularisation, and the weight attached to the positive (ADR) class.

4 Hyperparameter Tuning

Hyperparameters for our models were chosen based on the ADR class F1 score on the validation set.

4.1 Tokenising Tweets

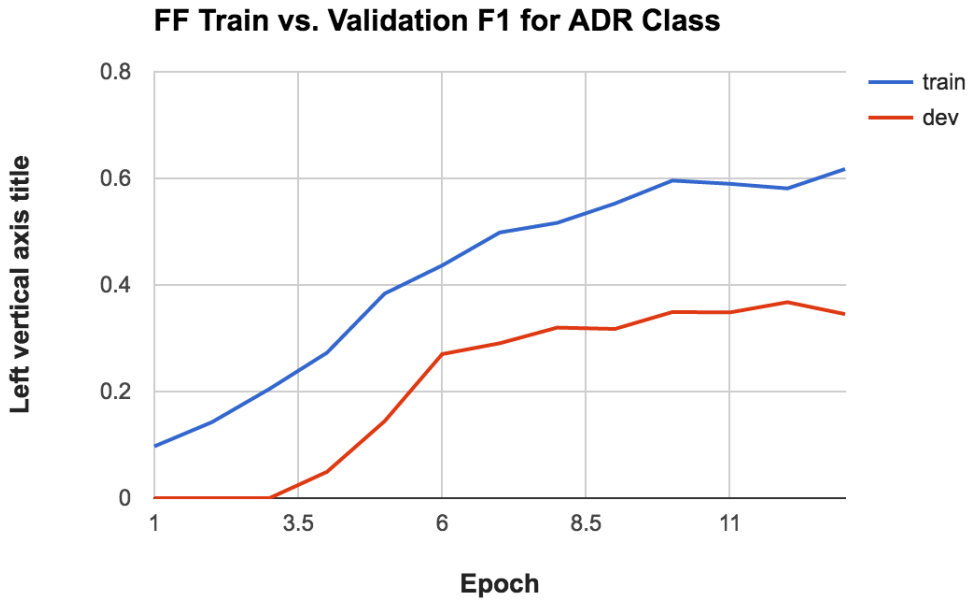
Due to the highly informal nature of our data, different methods of tokenisation were tested across all models for best results against the validation set. The best performance was achieved by keeping misspellings and embellishments, and merely removing special characters directly attached to words. For example, 'coooool' and 'cool' were kept as separate, valid words; '#xanax' was converted to 'xanax'. The plot below compares the best performance of the FF network using tokenisation wherein words with special characters were considered distinct (i.e., '#xanax' and 'xanax' considered distinct), and wherein words had special characters removed (i.e., '#xanax' was converted to 'xanax').



5 Results

5.1 Evaluation

The manually labelled tweets serve as the ground truth for all of our models. We evaluate these models based on precision, recall, and F1 score.



5.2 Regularisation

L2-regularisation was used in both RNN and CNN models, and dropout was used in all neural net models. We tested more aggressive λ values and dropout rates were used after noticing that train and validation set ADR class F1 scores diverged quickly after training ADR class F1 broke approximately 0.65. We performed a simple grid search over several L2 values and dropout rates for the final models of each type. An interesting observation is that despite more agressive regularisation, the divergence of train and validation F1 scores did not improve. After some exploration, we hypothesise that this is likely due to the high level of non-overlap between the train, validation, and test set vocabularies. Our final models use dropout rates of 90% for the CNN model, and 70% for the FF net.

The best results achieved by our models are displayed in the table below, along with the state of the art results from Sarker et al.

5.3 Loss Function

The loss function we used was a standard cross-entropy. Our group implemented batch stochastic gradient descent (batch size of 25). The highest validation ADR Class F1 was typically reached before epoch 10, and most models' losses stagnated once training ADR Class F1 reached approximately 0.65.

Table 2: Validation Set Best Performance by Model

	SotA	LR+WP	FF+WP	WindowedRNN+GloVe	CNN+GloVe
Precision	N/A	0.446	0.449	0.165	0.363
Recall	N/A	0.280	0.349	0.226	0.373
F1	0.592	0.344	0.392	0.19	0.368

The parameters that led to the best validation set performance were then saved and used against the test set described earlier in this paper. The top performing models were the FF and CNN models:

Table 3: Test Set Best Performance of Top 2 Models

	FF+WP	CNN+TweetStart
Precision	0.461	0.338
Recall	0.410	0.447
F1	0.434	0.385

6 Discussion and Conclusion

All models performed relatively poorly against the SotA model implemented by Sarker et al., whose group implemented a conditional random field model (CRF). While CRFs and other graphical models are able to encode priors and domain knowledge, there is no analogue for these priors in the intermediate layers of the neural net models implemented in this study. This could contribute to the discrepancy between Sarker's and our group's results.

Additionally, there were a large number of common words not present in the GloVe data set. Our custom Word Presence (WP) vectors outperformed the GloVe vectors in all models wherein the two embedding types were directly compared (i.e., LR and FF).

Though our models did not outperform the SotA, we are hopeful that further exploration of the hyperparameter space can bring the neural network models to comparable performance levels to the SotA.

Acknowledgments

We would like to thank the CS224D teaching team for their time and support, as well as the Diego Lab for their annotated Twitter data set.

References

[1] Iyyer, Mohit, Varun Manjunatha, Jordan Boyd-Graber, and Hal Daum Iii. "Deep Unordered Composition Rivals Syntactic Methods for Text Classification." Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers) (2015): n. pag. Web.

[2] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. GloVe: Global Vectors for Word Representation.

[3] OConnor, Karen et al. Pharmacovigilance on Twitter? Mining Tweets for Adverse Drug Reactions. AMIA Annual Symposium Proceedings 2014 (2014): 924933. Print.

[4] Sarker, Abeed, and Graciela Gonzalez. Portable Automatic Text Classification for Adverse Drug Reaction Detection via Multi-Corpus Training. Journal of biomedical informatics 53 (2015): 196207. PMC. Web. 16 May 2016.