HHS Public Access

Author manuscript

Proceedings (IEEE Int Conf Bioinformatics Biomed). Author manuscript; available in PMC
2018 July 03.

Published in final edited form as:

Proceedings (IEEE Int Conf Bioinformatics Biomed). 2017 November ; 2017: 1154–1159. doi:10.1109/

BIBM.2017.8217830

Deep Gramulator: Improving Precision in the Classification of Personal Health-Experience Tweets with Deep Learning

Ricardo A. Calix,

Purdue University Northwest, Hammond, USA

Ravish Gupta,

Purdue University Northwest Hammond, USA

Matrika Gupta, and

Purdue University Northwest, Hammond, USA

Keyuan Jiang

Purdue University Northwest Hammond, USA

Abstract

Health surveillance is an important task to track the happenings related to human health, and one of its areas is pharmacovigilance. Pharmacovigilance tracks and monitors safe use of pharmaceutical products. Pharmacovigilance involves tracking side effects that may be caused by medicines and other health related drugs. Medical professionals have a difficult time collecting this information. It is anticipated that social media could help to collect this data and track side effects. Twitter data can be used for this task given that users post their personal health related experiences on-line. One problem with Twitter data, however, is that it contains a lot of noise. Therefore, an approach is needed to remove the noise. In this paper, several machine learning algorithms including deep neural nets are used to build classifiers that can help to detect these Personal Experience Tweets (PETs). Finally, we propose a method called the Deep Gramulator that improves results. Results of the analysis are presented and discussed.

Keywords

natural language processing; deep learning

I. Introduction

Health surveillance is an important task to track the happenings related to human health, and one of its areas is pharmacovigilance. Pharmacovigilance tracks and monitors safe use of pharmaceutical products. Pharmacovigilance involves tracking and monitoring side effects that may be caused by medicines and other health related drugs. Medical professionals have a difficult time collecting this data because there are limited channels of communication between users of medical drugs and the people responsible for health surveillance. It is anticipated that social media could help to collect this data. Twitter data can be used for this task given that users post their personal health related experiences on line. One problem with twitter data, however, is that it contains a lot of noise. Therefore, an approach is needed to

remove the noise and obtain the relevant data from the tweets. Additionally, the collected data from Twitter, because of the noise, contains a very high imbalance where most tweets are irrelevant for the task. In this paper, several machine learning algorithms including deep neural nets are used to build classifiers that can help to detect this data. Here, precision is the metric of interest and not accuracy given that the goal is to automatically obtain relevant personal health experience tweets (PET) from social media.

Because data annotation is expensive, it is preferable to have high precision over high recall. The goal is to collect as many PET tweets as possible and to decrease the amount of Non-PET tweets that will need to be annotated. Therefore, a low recall score is acceptable. Finally, we propose a method called the Deep Gramulator that improves results. Results of the analysis are presented and discussed in section 4.

II. Literature Review

A. General

Health surveillance or pharmacovigilance is an important issue that must be addressed to reduce mortality rates. Pharmacovigilance deals with keeping track of the use of prescription drugs by patients. Prescription drugs are commonly tested with limited conditions in clinical trials before they can be brought to market and such testing may not reveal all the adverse drug reactions. According to [1], more than 90% of adverse drug reactions are under-reported. Social media could be a way to obtain this information from patients. Some recent work has shown promise that social media could be used for pharmacovigilance. The most common approach to extract this data such as effects from twitter text is to use string matching and lexical approaches. In [2], for instance, the authors developed a machine learning based method to extract adverse drug reaction mentions from Twitter data. One disadvantage of their analysis is that the data set was small. They only annotated 1340 tweets for training and 444 tweets for testing. Obtaining annotated and relevant tweets is one of the biggest challenges in this area and is one of the biggest motivators for the methodology presented in this paper. The authors in [2] addressed the issue of data annotation by collecting almost 397,729 drug related un-labeled tweets. They used the unlabeled tweets to perform word embeddings [4]. Once the word embeddings were created, they used K-means clustering to build “n” number of clusters (such as 150 clusters). Each one of these clusters represents a subset of words from the word embeddings. Out of these clusters, 7 were selected to be used as features. Each cluster represented a set of semantically related words that were searched for in the tweets. They used this approach to semi-annotate the tweets. One problem with collecting un-annotated tweets in this way is that they still contain a lot of noise and may not be very relevant.

In the work presented in [3] it has been noted that very few of all downloaded tweets are relevant and that the majority are noisy and irrelevant tweets. Therefore, methods for automatic detection of personal experience tweets are needed. In general, the approaches to obtain these tweets are unsupervised methods and supervised methods. Unsupervised methods can be advantageous because no annotation is needed. However, there are disadvantages because the methods cannot be very specific. One big advantage with unsupervised methods is that one can leverage very large amounts of data. The following

works have explored unsupervised approaches [5, 6, 7, and 10]. In [5] they used unsupervised approaches to model dialogue acts from twitter data. The method discovers dialogue acts by clustering raw utterances in tweets. In [10] the authors used an unsupervised framework to explore events from tweets. They implemented a pipeline for filtering, extraction, and categorization. To remove noisy tweets, they used a filtering step that exploits a lexicon-based approach to separate tweets that are event-related from those that are not event-related. Evaluation of their approach on a set of 60 million tweets achieved a precision of 70%. In [7] they used the power of newer deep learning approaches such as word embeddings to perform clustering based on the co-occurrence of words in tweets. The method in that work learns feature representations from tweets that are associated with hashtags. This helps them to organize the documents by hash tag. The analysis was done on up to 5.5 billion words and discovered 100,000 hashtag sets. Finally, a more detailed survey of various other unsupervised approaches can be found in [6].

One of the downsides of unsupervised methods is that they can only discover simple clusters based on the most frequent features in the clusters. For more specific tasks like finding higher level semantics in the tweet or personal experience text, a trained model with annotated data might be required. Supervised methods can be more specific in the classification task but are very expensive because the data needs to be annotated by humans. Therefore, a method to improve annotation by filtering out noisy tweets and discovering relevant tweets is needed. Most approaches (such as in [9]) will take a lexicon and syntactic based approach to extract features from tweets to perform classification. The big companies like Facebook, Google, and others have been successful in using supervised approaches because they have lots of labeled data given that users annotate messages and texts by rating their entries in these sites. The work in [8] provides a good overview of definitions, trends and challenges in detecting events in social media. Because of these issues, a lot less data can be used. Supervised methods are especially problematic when the data is highly imbalanced. Additionally, collecting enough annotated samples to build models that generalize well to the entire sample population is difficult. The work presented in this paper tries to address some of these issues and discusses some of the major challenges.

B. Grammulator

One technique in particular that has had a lot of success to extract features for supervised machine learning is called the gramulator [11]. The gramulator is a feature extraction technique used particularly in natural language processing. The main idea is that, for a 2 class problem, you want to extract features (e.g. words) that are very frequent in 1 class but infrequent in the other. This helps to better discriminate between the classes. The downside of this approach, however, is that it needs a lot of annotated or labeled data to extract the grams or words from each class that are infrequent in the opposite class. If the grams are representative of the entire population; then, it can be expected that a classifier will have good performance in the classification task measured.

III. Methodology

A. General

In this paper several supervised machine learning methods are studied to develop a classifier to detect PET tweets. Tenfold cross validation was used for the analysis on the training set. Additionally, an independent separate test set was used to measure how well the models generalize to the overall population. A total of 12,559 tweets were annotated by two nurses and one graduate student majoring in biology to be used for training. An additional separate test set of 3,156 samples was also annotated. The training data set of 12,559 samples had 2,021 PET tweets and 10,538 non-PET tweets. The test of 3,156 samples consisted of 622 PET tweets and 2,534 non-PET tweets.

The features used consisted of 20 features extracted from the tweet itself or the metadata of the tweet. Examples of some of the features include: count of pronouns in a tweet, most frequent terms in one class that are infrequent in the other class, count of URLs in a tweet, etc. For a more complete description of the features see [3].

For the analysis, deep neural nets of 2, 3, and 4 layers are used as well as other more traditional machine learning techniques like SVM, logistic regression, KNN, decision trees, etc. The deep neural nets were implemented in Tensorflow. The architectures consisted of from 2 to 4 hidden layers. The input layer had 20 neurons. The hidden layers also consisted of 20 neurons each and the output layer consisted of 2 neurons. The neural networks were quick to process and no more than 1000 epochs were needed per architecture.

B. Deep Gramulator

The proposed deep gramulator approach is a combination of the well proven principles of the gramulator (as previously defined) with the abilities of word embedding techniques such as word2vec [12]. As previously defined, the gramulator is a feature extraction technique used particularly in natural language processing and it requires annotation of data. It becomes challenging to obtain the optimal sets of grams when we do not have that much annotated data. Here, word embeddings can help. Word embeddings can use the co-occurrence of words in a large data set of documents to create a vector space that can represent some aspects of this relationship.

Combining these two frameworks, the gramulator and word embeddings, we can naturally develop the deep gramulator which can be used to find grams from large amounts of un-labeled data by using a small set of annotated text. The algorithm creates the vector space with both text sets (labeled and un-labeled) and then proceeds to find the closest grams from un-labeled texts to the grams of labeled texts. In this case, from a set of originally labeled grams, an expanded set can be obtained. For example, given a 2 class problem, we can use a small set of grams that are very frequent in class 1 but infrequent in class 2, to find more terms in the unlabeled data that might be related to class 1. As a result, you can obtain an expanded set of grams.

In the rest of this section, the deep gramulator algorithm will be discussed. The first part of the algorithm for the deep gramulator uses the Porter stemmer to shorten words. The Porter

stemmer can take several words such as happiness, happy, happiest, etc. and shorten them to their root or stem (e.g. happ). This is a very useful thing to do as it can result in more matches when the words are being compared.

The next part of the algorithm focuses on reading the word vectors from the vector space model. This assumes, of course, that the set of text documents has already been processed through word2vec. Here, each word is represented as a vector where the first column is the word and the columns 1 (the second) through 128 (the 129th) are the word2vec features that represent the word.

Once the vectors have been extracted, the next step is to perform the similarity calculation between all the vectors (i.e. all the words). Here, the most efficient approach is to perform a multiplication of all the word vectors at the same time.

The key aspect is that the word matrix M (i.e. Words) is multiplied by the transpose of itself. This operation can be expressed as follows

$$s = M \cdot M^T \quad (1)$$

and can be further expressed for each pair of vectors (a and b) as a cosine similarity metric given as

$$a \cdot b = \|a\| \|b\| \cos\theta \quad (2)$$

The next step in the process is to obtain the list of words (seed list) that have already been annotated as being either from one class or the other.

The next step in the algorithm is to use the selected words per class to find the closest matches in the matrix M . This will result in 2 matrices (1 per class assuming a 2 class problem) that contain the vectors of the words that are frequent in 1 class while infrequent in the other.

Finally, the algorithm takes the top 10 largest (i.e. most similar) words. Therefore, the function can obtain the 10 most similar words to a given annotated word per class. The following algorithm expresses the idea.

```
top_n = 10
result_auto_train_yes = pd.DataFrame({
    n:
    selected_rows_auto_yes_train.T[col].nlargest(top_n).index.tolist()
    for n, col in
    enumerate(selected_rows_auto_yes_train.T)
}).T
```

As a result, we now have the list of words that are closest to the annotated words per class. We can then write these out to a file or use them in the next step of our classifier.

IV. Analysis and results

A. General

The results of the analysis are presented and discussed in this section. Figure 1 presents the challenges that must be addressed when dealing with data of this nature. After performing principle component analysis (PCA) it can be seen that there is high overlap in the classes and that the data is difficult to separate. A linear classifier can only separate a small portion of the PET tweets from the non-PET tweets. This can help with precision but affects recall.

It can be seen that under 10-fold cross validation of the training set the classifier is able to discern with some precision the PET samples from non-PET samples. One of the challenges, however, is that many PET tweets cannot be detected because they overlap with the noisy data. In Figure 1, the x axis is for PCA1 and the y axis is for PCA2.

B. Support Vector Machines(SVM)

SVM methods can sometimes build better non-linear classifiers because of their ability to project data to higher dimensional spaces.

Figure 2 shows the SVM classifier building a non-linear separation line on the PCA data. The classifier does really well on the training data but not well on the independent test set of 3,156 samples (see below). Using a gamma of 0.001 and cost of 32 on an RBF kernel, SVM achieves very high ROC area values greater than 90% for each fold. However, there is a possibility that this model may over fit and does not apply well to the independent test set. Table I presents the results of the confusion matrix when an SVM classifier is used. The number of samples predicted as PET is small. Although recall is low, precision is high in this case.

C. Deep Learning

Figure 3 presents the accuracy plot per epoch as the deep neural net is being trained. These results are high because of the class imbalance in the data.

The classifier favors the dominant noisy non-PET class and that is reflected in accuracy using the train set. Figure 4 shows that the precision is lower but still good when using the training set with 10 fold cross validation.

The deep net takes a few epochs to learn how to detect and classify the samples. After about 200 epochs, the deep net of 2 hidden layers stabilized and maintained a consistent precision score. Here HL stands for hidden layers. Tables 2 and 3 present the confusion matrices for the deep neural nets. In general, the 4 layer deep net had more trouble converging and had slightly worse results than the 2 layer deep neural net. It is possible that a not very deep net may be the best choice given that there are only 20 input features.

D. Classifier Comparison

Table 4 presents the results of comparing the different supervised learning classifiers when using 10 fold cross validation on the train set. It can be seen that the neural nets have consistent F1 measure scores. SVM has the worst recall scores given that it does very well

on precision. This is as a result of using an RBF kernel with parameters gamma equal to 0.001 and cost of 32.

E. Precision over 10-Fold Cross Validation on Train Set

The precision score is the main interest of this work. The results are presented separated out by fold in Table 5. On the train set analysis, it can be seen that SVM does best.

Deep nets do worse than the SVM but are still good. Interestingly, the simple linear logistic regression model has good precision as well. The methods with the lowest precision include decision trees and the KNN classifier.

F. Analysis on Independent Test Set of 3156 samples

The classifiers proposed in this work are to be used to detect PET tweets from social media in Twitter. Therefore, it must be tested on an independent set of data collected separately from the data collected for training. This is very challenging since the data in Twitter is very massive and is always changing.

Not many studies choose to use a test set that is very different from the train set. However, here we have chosen to take this approach to measure how well the Deep Gramulator generalizes to the population. The scores are expected to be lower for this test set.

The task is very challenging as can be seen in Table 6. Even though some classifiers performed well on the training set using 10 fold cross validation, the performance drops when an independent test set is used.

This may indicate that not enough data has been collected and that the collected data so far does not yet represent all the diversity in the Twitter population data. However, once enough data is obtained, it is anticipated that the classifiers will perform nearer the metrics obtained during 10-fold cross validation of the train set.

Table 7 presents the confusion matrix when using a deep neural net with 2 hidden layers on the test set. It can be seen that out of 1,339 predicted PET tweets, only 347 are actually PET tweets. Figure 5 presents a plot of the precision per epoch as a deep net with 4 hidden layers is training. After about 50 epochs the precision begins to stabilize around 0.25. No more than 1,000 epochs were used in training the deep neural nets.

G. Independent Test Set Analysis with Deep Gramulator

In this section we re-run the data analysis using our proposed Deep Gramulator algorithm.

As can be seen, the results have improved with the deep gramulator. This shows evidence that the proposed deep gramulator approach is discovering new terms that can help to discriminate between the 2 classes.

V. conclusion

In conclusion, precision is good when using 10 fold cross validation on the train set of 12,559 samples. When applied to the independent test set of 3,156 samples, the classifiers do

not do as well with regards to precision. The main problem is most likely related to the small amount of data that has been collected so far. The limited size may not allow the models to generalize fully to the entire population. The analysis of the proposed Deep Gramulator method shows promise for this approach to improve precision in the detection of PET tweets.

Acknowledgments

The authors wish to thank Dustin Franz for data collection, Bridget Swindell, Mary Stroud and Cecelia Lai for annotating the tweets, as well as Dr. Gordon Bernard for guidance. This work was supported in part by the National Institutes of Health grant 1R15LM011999-01.

References

1. Hazell L, Shakir SAW. Under-Reporting of adverse drug reactions. *Drug Safety*. 2006; 29:385–396. [PubMed: 16689555]
2. Nikfarjam A, Sarker A, O'Connor K, Ginn R, Gonzalez G. Pharmacovigilance from social media: mining adverse drug reaction mentions using sequence labeling with word embedding cluster features. *Journal of the American Medical Informatics Association*. 2015; 0:1–11.
3. Jiang, K., Calix, R., Gupta, M. Proceedings of the 15th Workshop on Biomedical Natural Language Processing. Berlin, Germany: Association for Computational Linguistics; Aug 12, 2016 Construction of a personal experience tweet corpus for health surveillance; p. 128-135.
4. Turian J, Ratinov L, Bengio Y. Word representations: a simple and general method for semi-supervised learning. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*. Jul.2010 :384–394.
5. Ritter A, Cherry C, Dolan B. Unsupervised modeling of Twitter conversations. In *Proceedings of HLT '10 Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*. 2010:172–180.
6. Atefeh F, Khreich W. A survey of techniques for event detection in Twitter. *Computational Intelligence*. 2013; 0(0)
7. Weston J, Chopra S, Adams K. #TagSpace: semantic embeddings from hashtags. *Empirical Methods in Natural Language Processing (EMNLP)*. Sep 4.2014
8. Panagiotou N, Katakis I, Gunopulos D. Detecting Events in Online Social Networks: Definitions, Trends and Challenges. *Solving Large Scale Learning Tasks Challenges and Algorithms*, Volume 9580 of the series *Lecture Notes in Computer Science*. Jul 03.2016 :42–84.
9. Ginn R, Pimpalkhute P, Nikfarjam A, Patki A, O'Connor K, Sarker A, Smith K, Gonzalez G. Mining Twitter for adverse drug reaction mentions: a corpus and classification benchmark. In *Proceedings of the 4th workshop on building and evaluating resources for health and biomedical text processing*. 2014:1–8.
10. Zhou D, Chen L, He Y. An Unsupervised Framework of Exploring Events on Twitter: Filtering, Extraction and Categorization. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*. 2015:2468–2474.
11. McCarthy P, Watanabe S, Lamkin T. The Gramulator: A Tool to Identify Differential Linguistic Features of Correlative Text Types. IGI-Global. 2012
12. Mikolov T, Chen K, Corrado G, Dean J. Efficient estimation of word representations in vector space. *ICLR Workshop*. 2013

Fig. 1.
Logistic regression classifier.

Fig. 2.
SVM classifier

Fig. 3.
Accuracy per epoch Deep H2 on train set.

Fig. 4.
Precision per epoch Deep (HL=2)

Fig. 5.
Deep Net (HL=4) Precision per epoch on 3156 Test set

TABLE I

Confusion Matrix with SVM classifier

Predicted	Non-PET	Predicted PET	Predicted Total
Actual Non-PET	1,035	8	1,043
Actual PET	124	75	199
Actual Total	1,159	83	1,242

Author Manuscript

Author Manuscript

Author Manuscript

Author Manuscript

TABLE II

Confusion matrix with deep net (2 hidden layers)

	Predicted Non-PET	Predicted PET	Predicted Total
Actual Non-PET	1,014	18	1,032
Actual PET	101	111	212
Actual Total	1,115	129	1,244

Author Manuscript

Author Manuscript

Author Manuscript

Author Manuscript

TABLE III

Confusion Matrix with deep net (4 HL) – train set

	Predicted Non-PET	Predicted PET	Predicted Total
Actual Non-PET	974	55	1,029
Actual PET	84	130	214
Actual Total	1,058	185	1,243

Author Manuscript

Author Manuscript

Author Manuscript

Author Manuscript

TABLE IV

Machine learning Comparison on train set

Method	Accuracy	Recall	F1
KNN	0.88	0.50	0.58
Decision Trees	0.86	0.57	0.57
Random Forest	0.89	0.48	0.6
Logis. Regress.	0.90	0.53	0.64
SVM (rbf)	0.89	0.37	0.53
Deep Net (hl=2)	0.89	0.51	0.62
Deep Net (hl=3)	0.89	0.52	0.62
Deep Net (hl=4)	0.89	0.52	0.62

TABLE V

Classifier precision per fold on train set

Method	1	2	3	4	5	6	7	8	9	10
KNN	66	75	70	70	72	65	65	73	73	69
Decision Trees	58	55	52	52	59	53	61	58	58	60
Random Forest	76	84	78	76	88	73	72	79	77	82
Logistic Regression	80	81	81	78	87	81	79	84	86	84
SVM (rbf)	91	90	90	85	92	95	91	91	96	90
Deep Net (hl=2)	86	79	75	80	81	78	81	83	80	79
Deep Net (hl=3)	78	79	78	77	80	80	78	78	77	73
Deep Net (hl=4)	78	86	74	75	76	76	70	80	71	75

TABLE VI

Metrics on Independent 3156 Test Set

Method	Accuracy	Precision	Recall	F1
KNN	0.57	0.23	0.53	0.33
Decision Trees	0.57	0.25	0.60	0.35
Random Forest	0.59	0.24	0.51	0.33
Logis. Regress.	0.58	0.23	0.52	0.32
SVM (rbf)	0.55	0.18	0.36	0.24
Multilayer Percept.	0.60	0.26	0.59	0.36
Deep Net (hl=2)	0.60	0.25	0.55	0.35
Deep Net (hl=3)	0.58	0.24	0.54	0.33
Deep Net (hl=4)	0.57	0.25	0.60	0.35

TABLE VII

3156 Deep net (HL=2) confusion Matrix

Predicted	Non-PET	Predicted PET	Predicted Total
Actual Non-PET	1,542	992	2,534
Actual PET	275	347	622
Actual Total	1,817	1,339	3,156

Author Manuscript

Author Manuscript

Author Manuscript

Author Manuscript

TABLE VIII

Deep gramulator approach on 3156 Test Set

Method	Precision (Previous)	Precision Deep Gramulator
KNN	0.23	0.35
Decision Trees	0.25	0.31
Random Forest	0.24	0.34
Logis. Regress.	0.23	0.34
SVM (rbf)	0.18	0.31
Multilayer Percept.	0.26	0.32