AUTOMATED DETECTION OF MISINFORMATION IN TWEETS ABOUT COVID-19

A PREPRINT

Caitlin Moroney *
Department of Mathematics and Statistics
American University
Washington, DC 20016
cm0246b@american.edu

November 28, 2020

ABSTRACT

Enter the text of your abstract here.

***K*eywords** COVID-19 · coronavirus · NLP · machine learning · misinformation

1 Introduction

Insert introduction text here.

The rest of the paper is organized as follows: Section 2 discusses our methodology; Section 3 presents the results of our experiments; Section 4 discusses our results and plans for future work.

2 Methodology

This paper explores a series of methods which seek to exploit linguistic features in raw text data in order to perform the automated detection of unreliable tweets. The experiments follow the same general framework with certain implementation details tweaked for each experiment. The first step in this framework is the application of NLP featurization methods to the raw tweet text. While different featurization methods are compared, all methods involve the use of NLP tools to represent the text with numeric features. Subsequently, latent variable methods are employed to reduce the dimensionality of the resulting X feature matrix (as well as to uncover latent variables). The latent variables are then used in the classification task. Finally, we evaluate the classification algorithms paired with the featurization methods with respect to performance and explainability. We use the typical performance metrics, including accuracy, F-score, precision, recall, and ROC-AUC. We use LIME to evaluate local explainability for non-interpretable methods. Furthermore, we present a new explainability framework for latent variable methods as well as a new explainability metric. Below, we explore in greater detail the methods used for featurization, latent variables, classification, and evaluation of explainability.

2.1 NLP featurization

In order to obtain features from the raw tweet text, we first employed standard preprocessing, to include removing stop words and punctuation as well as lemmatizing words. The two approaches we pursued involved (1) Bag-of-Words and (2) word embeddings, each followed by latent variable methods.

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

2.1.1 Bag-of-Words

- Bag-of-Words + topic modeling
  - Vectorization: raw counts, tf-idf, binary
  - PCA + ICA

After constructing the Bag-of-Words matrix, we employed topic modeling to reduce the dimensionality of the Nxp matrix.

2.1.2 Word embeddings

To create word embeddings, we used the word-context co-occurrence matrix which, unlike the Bag-of-Words method, is able to incorporate information about context from the raw text data. We trained the embeddings on the full set of 560 tweets. With the Bag-of-Words methods, we are able to encode information about words' presence in a given document; the co-occurrence approach improves upon this by allowing us to look at word usage with respect to the presence of other words. We constructed a word-context co-occurrence matrix using the entire vocabulary for both target terms and context terms. In other words, the matrix was symmetric. We incorporated a number of hyperparameters related to the construction and subsequent transformation of this matrix, including context window size, the use of raw counts or variations on the Pointwise Mutual Information (PMI), and Laplace smoothing. We also included "<START>" and "<END>" tokens at the beginning and end of each tweet.

- Context window size

Window size refers to the number of tokens before and after the target word are scanned for context words. According to some sources, a window size of four or larger captures semantic representations of the words, whereas smaller windows capture more syntactic representations.

- Raw counts vs PMI (or PPMI)
- Shifted vs unshifted PMI (or PPMI)
- Laplace smoothing
- Use of start & end tokens

Latent variable methods were subsequently applied to the word embeddings in order to reduce the dimensionality.

Finally, we averaged over the word embeddings for the words in each tweet to obtain a single vector representation for each tweet.

Overall, it seemed that larger context windows prevailed - a window size of 15 +/- performed the best (I tried window sizes of 1, 2, 4, 6, 10, 15, and 20). I find this interesting because tweets are such short posts (perhaps 30 words encompasses the entirety of the tweet for most tweets). Incorporating add-one Laplace smoothing and shifted [P]PMI decreased performance across all metrics (accuracy, precision, recall, ROC AUC, and F1 score). Interestingly, PMI often outperformed not only raw frequencies but also PPMI (positive PMI). However, with other optimal hyperparameters held constant, there was virtually no difference in performance between PMI and PPMI. Optimal text cleaning included removing special characters which were not punctuation (parentheses, question marks, exclamation points, periods, commas, hyphens, colons, and semicolons) or alphanumeric characters (letters or numbers), converting all text to lowercase, removing stop words, and lemmatizing words (using NLTK's WordNetLemmatizer aided by NLTK's part-of-speech tagger).

The best results included ROC AUC of 0.94, accuracy of 0.89, F1 score of 0.90, precision of 0.86, and recall of 0.94. These scores are all higher than the best results from last week and were obtained using the same nested CV procedure (3 folds for inner loop; 5 folds for outer loop). NOTE: These should be replaced with the correct numbers.

- BERT embeddings from pre-trained model

2.2 Latent variable methods

In order to reduce the dimensionality of the feature matrix X, we employed latent variable methods. Specifically, we applied Principal Component Analysis (PCA) followed by Independent Component Analysis (ICA) to both the Bag-of-Words matrix and the word-context co-occurrence matrix.

- Explain PCA
- Explain ICA
- Topic modeling application explanation
- Word embeddings application explanation

The number of components is also up for question - it seems that people use values anywhere from 50 to 1,000.

2.3 Classification

- One-class SVM
- Binary SVM

2.4 Evaluation

- Performance
- Explainability
  - LIME (local)
  - ICA
    - global
    - local
  - Metric I came up with

Penalty:

$$\frac{1}{N}\sum_{i=1}^{N}\frac{1}{T_i}\sum_{j=1}^{T_i}1_A(w_j)-1_B(w_j)$$

No Penalty:

$$\frac{1}{N}\sum_{i=1}^{N}\frac{1}{T_i}\sum_{j=1}^{T_i}1_A(w_j)$$

3 Results

Report evaluation metrics for...

- One-class SVM
- Binary SVM

4 Discussion

- Interpret results
- Future work
  - Different ICA algorithm (not FastICA)
  - Multiple ICA runs + take most representative of those
  - Better ICA explainability tool?
  - Better explainability metric?
  - Other classifiers (e.g., neural nets)
  - Incorporate other features:
    - Part-of-speech tag counts
    - Punctuation counts
    - Use of all-caps
    - Sentiment analysis

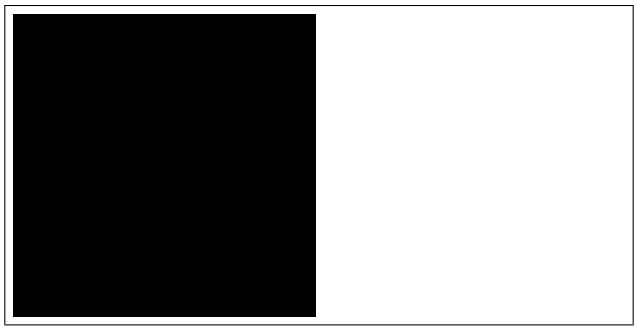


Figure 1: Sample figure caption.

Misc. equation.

$$\xi_{ij}(t) = P(x_t = i, x_{t+1} = j|y, v, w; \theta) = \frac{\alpha_i(t) a_{ij}^{w_t} \beta_j(t+1) b_j^{v_{t+1}}(y_{t+1})}{\sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i(t) a_{ij}^{w_t} \beta_j(t+1) b_j^{v_{t+1}}(y_{t+1})}$$

Paragraph Misc. text.

5 Examples of citations, figures, tables, references

Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Donec odio elit, dictum in, hendrerit sit amet, egestas sed, leo. Praesent feugiat sapien aliquet odio. Integer vitae justo. Aliquam vestibulum fringilla lorem. Sed neque lectus, consectetuer at, consectetuer sed, eleifend ac, lectus. Nulla facilisi. Pellentesque eget lectus. Proin eu metus. Sed porttitor. In hac habitasse platea dictumst. Suspendisse eu lectus. Ut mi mi, lacinia sit amet, placerat et, mollis vitae, dui. Sed ante tellus, tristique ut, iaculis eu, malesuada ac, dui. Mauris nibh leo, facilisis non, adipiscing quis, ultrices a, dui.

some text (Kour and Saabne 2014b, 2014a) and see Hadash et al. (2018).

The documentation for `natbib` may be found at

`http://mirrors.ctan.org/macros/latex/contrib/natbib/natnotes.pdf`

Of note is the command `\citet`, which produces citations appropriate for use in inline text. For example,

```
\citet{hasselmo} investigated\dots
```

produces

Hasselmo, et al. (1995) investigated…

`https://www.ctan.org/pkg/booktabs`

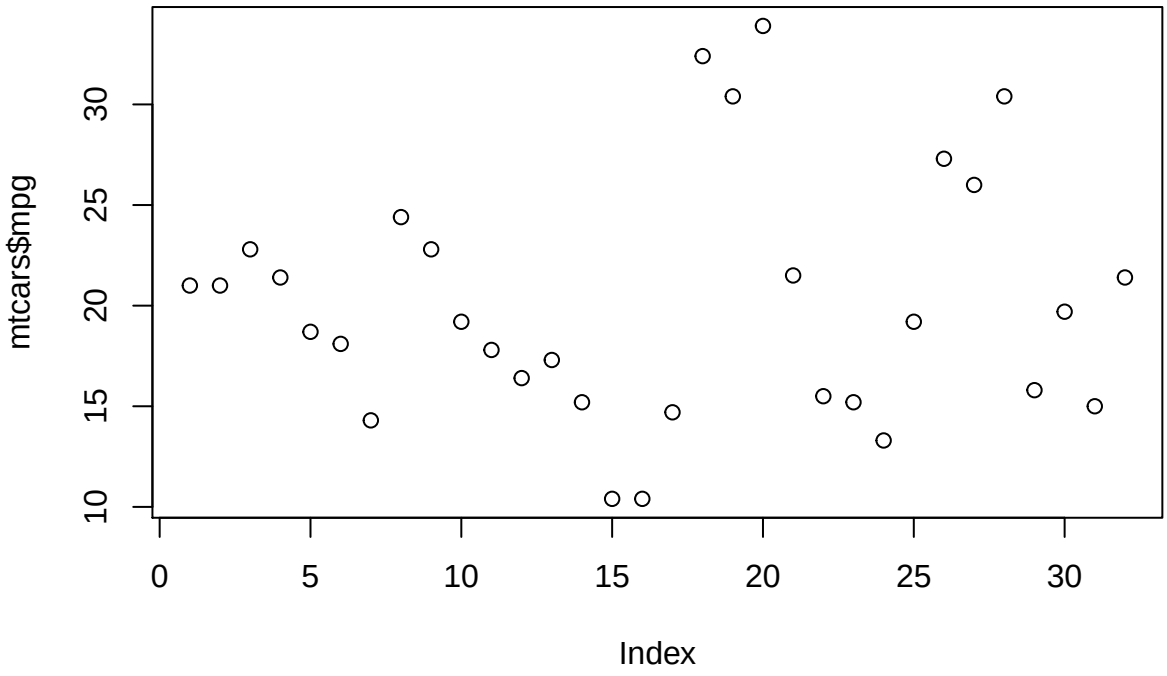
5.1 Figures

Misc. text. See Figure 1. Here is how you add footnotes. [^Sample of the first footnote.]

```
plot(mtcars$mpg)
```

Table 1: Sample table title

	Part	
Name	Description	Size (μm)
Dendrite	Input terminal	$\sim$100
Axon	Output terminal	$\sim$10
Soma	Cell body	up to 10^6



5.2 Tables

Misc. text.

See awesome Table~1.

References

Hadash, Guy, Einat Kermany, Boaz Carmeli, Ofer Lavi, George Kour, and Alon Jacovi. 2018. "Estimate and Replace: A Novel Approach to Integrating Deep Neural Networks with Existing Applications." *arXiv Preprint arXiv:1804.09028*.

Kour, George, and Raid Saabne. 2014a. "Fast Classification of Handwritten on-Line Arabic Characters." In *Soft Computing and Pattern Recognition (Socpar), 2014 6th International Conference of*, 312–18. IEEE.

———. 2014b. "Real-Time Segmentation of on-Line Handwritten Arabic Script." In *Frontiers in Handwriting Recognition (Icfhr), 2014 14th International Conference on*, 417–22. IEEE.