A Semi-Supervised Framework for Misinformation Detection on Social Media

Yashar Mansouri

DATA-642 – Advanced Machine Learning // American University, College of Arts and Sciences, Washington DC 20016

Detection of misinformation has been a hot topic in the semantic analysis branch of natural language processing. The common methods surrounding these methods have low computational efficiency since they treat language units as atomic representations, and the features are usually sparse and high dimensional. These representations do not consider context of the sentence nor different meaning of the same words used in different semantics.

A semi-supervised approach to create a feature set by using word embeddings and keeping their context into account is provided. These features were input in common machine learning frameworks and the classification metrics between true and unreliable tweets were compared.

Exploratory Data Analysis

- Tweets collected on COVID [2020 Q1-2]
- **Training**: 3792 True, 3794 Unreliable
  - After cleaning / deduplication >> 3626, 3135
  - 3k+ Indian
  - 2k+ Europe (UK)
  - 1.5k+ USA
- **Testing**: 280 True, 280 Unreliable
  - After cleaning / deduplication >> 277, 280

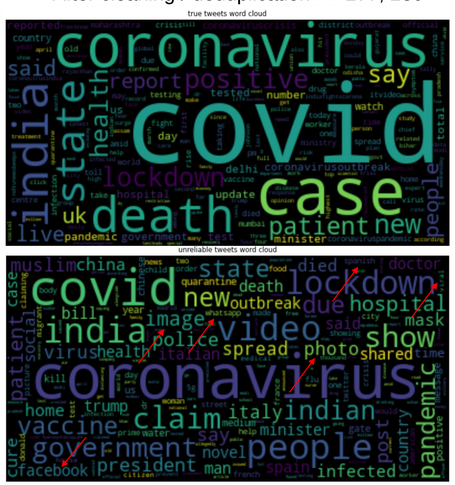

Figure 1. Word cloud representation words in true (above) and unreliable (bottom) tweets

Methods

Word2Vec model was separately applied on the true and unreliable tweets for both Continuous Bag of Words (CBOW) and SkipGram (SG) representations.

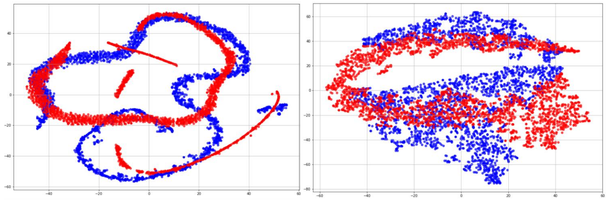

Figure 2. TSNE dimension reduced word embedding representations on shared words between the true (blue) and unreliable (red) tweets. CBOW model is shown on the left and SFG model is shown on the right.

By applying element-wise maximum between the embeddings of the shared words, their final representations were created. If a word was not shared, its own separate embedding was used. In order the represent the tweets, element-wise average of the word embeddings that were in a tweet were used.

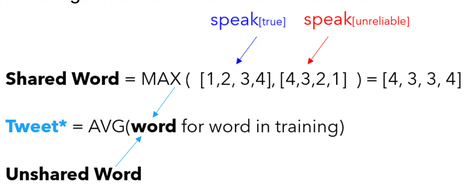

Figure 3. Tweet representation approach by using element-wise maximum on shared word embeddings and averaging

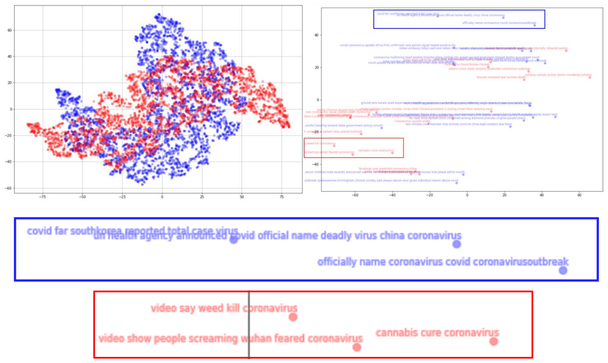

Figure 4. True (blue) and unreliable (red) tweet representations of CBOW embeddings

Results

Common machine learning models on two sets of training embeddings were used. The first set did not use shared embeddings from the true and unreliable category while the second set used the element-wise maximum of shared embeddings.

For both SG and CBOW embeddings, shared embeddings had significantly better results on validation (stratified 5-Fold) sets.

☑ Shared

Model	Accuracy	AUC	Recall	Prec.	F1
CatBoost Classifier	0.9042	0.9612	0.9366	0.8899	0.9126
Light Gradient Boosting Machine	0.9029	0.9619	0.9308	0.8922	0.9110
Extreme Gradient Boosting	0.9016	0.9611	0.9287	0.8918	0.9097
Linear Discriminant Analysis	0.8929	0.9496	0.9332	0.8749	0.9030
Extra Trees Classifier	0.8929	0.9529	0.9249	0.8809	0.9022
Gradient Boosting Classifier	0.8879	0.9525	0.9190	0.8772	0.8975
Ridge Classifier	0.8867	0.0000	0.9280	0.8688	0.8974
Random Forest Classifier	0.8863	0.9502	0.9024	0.8868	0.8944
Quadratic Discriminant Analysis	0.8683	0.9276	0.8636	0.8871	0.8751
Logistic Regression	0.8661	0.9292	0.8944	0.8605	0.8771
Ada Boost Classifier	0.8641	0.9341	0.8899	0.8608	0.8748
SVM - Linear Kernel	0.8619	0.0000	0.8788	0.8678	0.8711
K Neighbors Classifier	0.8521	0.9144	0.9477	0.8084	0.8725
Decision Tree Classifier	0.8129	0.8122	0.8220	0.8269	0.8243
Naive Bayes	0.7901	0.8846	0.7434	0.8450	0.7908

Model	Accuracy	AUC	Recall	Prec.	F1
Light Gradient Boosting Machine	0.9784	0.9978	0.9806	0.9790	0.9798
CatBoost Classifier	0.9773	0.9982	0.9785	0.9789	0.9787
Extreme Gradient Boosting	0.9765	0.9977	0.9775	0.9786	0.9780
Quadratic Discriminant Analysis	0.9752	0.9907	0.9796	0.9742	0.9769
Gradient Boosting Classifier	0.9739	0.9974	0.9751	0.9762	0.9756
Extra Trees Classifier	0.9726	0.9965	0.9761	0.9728	0.9744
Linear Discriminant Analysis	0.9710	0.9961	0.9834	0.9631	0.9731
Ridge Classifier	0.9706	0.0000	0.9848	0.9612	0.9728
Random Forest Classifier	0.9699	0.9965	0.9668	0.9766	0.9717
SVM - Linear Kernel	0.9695	0.0000	0.9768	0.9668	0.9715
Ada Boost Classifier	0.9671	0.9959	0.9682	0.9702	0.9692
Logistic Regression	0.9656	0.9952	0.9754	0.9608	0.9681
K Neighbors Classifier	0.9565	0.9872	0.9751	0.9454	0.9600
Decision Tree Classifier	0.9297	0.9294	0.9343	0.9343	0.9342
Naive Bayes	0.8438	0.9480	0.7630	0.9323	0.8390

Figure 5. Left table shows validation results from separate SG embeddings. Right table shows the classification metric improvements by using SG shared embeddings. Similar results were achieved by using CBOW as well.

Light Gradient Boosting Machine, or Linear Discriminant Analysis models were chosen due to their high metrics and low fitting times. They were tuned on the F1 metric and evaluated on the testing set. The models did not perform well on unseen data (65% acc vs. 97%) which can be due to missing words that appear in testing set.

CBOW - Shared

	precision	recall	f1-score	support
0	0.67	0.62	0.64	280
1	0.64	0.68	0.66	277
accuracy			0.65	557
macro avg	0.65	0.65	0.65	557
weighted avg	0.65	0.65	0.65	557

Separate

	precision	recall	f1-score	support
0	0.66	0.91	0.77	280
1	0.85	0.53	0.65	277
accuracy			0.72	557
macro avg	0.76	0.72	0.71	557
weighted avg	0.76	0.72	0.71	557

SG - Shared

	precision	recall	f1-score	support
0	0.65	0.61	0.63	280
1	0.63	0.66	0.64	277
accuracy			0.64	557
macro avg	0.64	0.64	0.64	557
weighted avg	0.64	0.64	0.64	557

Separate

	precision	recall	f1-score	support
0	0.65	0.88	0.75	280
1	0.81	0.53	0.64	277
accuracy			0.70	557
macro avg	0.73	0.70	0.69	557
weighted avg	0.73	0.70	0.69	557

Figure 6. Comparison of shared (left) and separate (right) embeddings on the test set

Conclusion and Future Directions

- Test set had **590+ words missing** in the training bag of words, which can lead to losing tweet context.
- As with most models, it is expected the unseen performance will improve as more data gets introduced into the training set.
- Models can focus on **region** specific words (India, UK, Delhi, etc.) or **hashtags** (#coronavirusoutbreak, …) (happens less in embedding representations), yet removing them can improve the results.
- Creating **separate embeddings** on missing words didn't help with testing accuracy since the representations were not the same as training.
- An intervening flexible model can be trained between the common words of testing and training embeddings and be used to change embeddings on missing words, so they become similar to training.

References

1. Spotting Misinformation On Social Media Is Increasingly Challenging, Forbes, Peter Suciu, Aug 2, 2021
2. Why do we use word embeddings in NLP?, Towards Data Science, Natasha Latysheva, Sep 10, 2019
3. NLP 101: Word2Vec — Skip-gram and CBOW, Towards Data Science, Ria Kulshrestha, Nov 24, 2019
4. Machine Learning Explainability vs Interpretability: Two concepts that could help restore trust in AI, KDnuggets, Richard Gall, Dec 2018
5. KI^2TE: Knowledge-Infused InterpreTable Embeddings for COVID-19 Misinformation Detection - William Shiao, Evangelos E. Papalexakis
6. The Case for Latent Variable vs Deep Learning Methods in Misinformation Detection: An Application to COVID-19 - Caitlin Moroney, Evan Crothers, Sudip Mittal, Anupam Joshi, Tulay Adali, Christine Mallinson, Nathalie Japkowicz, and Zois Boukouvalas
7. Five sources of bias in natural language processing - Dirk Hovy, Shrimai Prabhumoye

linkedin.com/in/yasharmansouri

github.com/ym88