Detecting Fake News Articles

Jun Lin*
Computer Science and Engineering
Louisiana State University
Baton Rouge, LA, USA
jlin21@lsu.edu

Glenna Tremblay-Taylor*
Computer Science
Keene State College
Keene, NH, USA
glenna.taylor@ksc.keene.edu

Guanyi Mou, Di You, Kyumin Lee
Computer Science
Worcester Polytechnic Institute
Worcester, MA, USA
{gmou,dyou,kmlee}@wpi.edu

***Abstract*—Fake news has been generated and widely spread although journalists and researchers created fact-checking websites (e.g., Snopes and PolitiFact) and analyzed characteristics of fake news. To fill this gap, in this paper we focus on developing machine learning models based on only text information in news articles toward automatically detecting fake news. In particular, we proposed a framework which extracts 134 features and builds traditional known machine learning models like Random Forest and XGBoost. We also propose a deep learning based model (LSTM with self-attention mechanism) to see which one performs better in the fake news article detection in both political news and celebrity news domains. In the experiments, we compare our models against 7 baselines. The results show that our XGBoost model improved 16.4% and 13.1% over the best baseline in terms of accuracy in both political news articles and celebrity news articles, respectively.**

***Index Terms*—Fake news detection, machine learning, long short term memory**

I. Introduction

Spreading rumors, misinformation and fake news have become one of important and serious problems in the society. Especially, "fake news" has become a popular term in the 2016 US presidential election and after the election. With the advancement of technology such as smart phones, voice-controlled devices (e.g., Alexa Echo devices and Google home devices), and smart watches, people can easily access online systems and are often exposed by fake news. Since so many news are generated everyday from the traditional main media, online social systems, personal broadcasting system like YouTube, recognizing and deciding which one is fake news or not becomes a hard and non-trivial problem.

To resolve the fake news problem, researchers analyzed characteristics of fake news [1]–[3]. In addition, fact-checking websites such as Snopes[1] and PolitiFact[2] have emerged. According to Report Lab's analysis in 2016, the number of fact-checking websites went up by 50% [4]. Despite of these efforts, fake news is still generated and widely disseminated. To complement the prior approaches against fake news, in this paper, we study how to build machine learning models which automatically identify fake news articles. Following the prior work [5], a fake news article is defined as a news article which contains intentionally false information.

TABLE I
Dataset.

News Article	Fake	True
PolitiFact	378 (42.3%)	516 (57.7%)
Gossip Cop	4,844 (23.0%)	16,213 (77.0%)
Total	5,222 (23.8%)	16,729 (76.2%)

In particular, we propose a framework which preprocesses each news article, extracts features and builds machine learning models. We build machine learning models based on traditionally known algorithms such as Logistic Regression, Support Vector Machine [6], and Random Forest, and propose and build a long short-term memory (LSTM) [7] with attention [8] based model, focusing on only text information of each news article. To evaluate performance of our models, we use a publicly available *FakeNewsNet* dataset [2], [5] which contains news articles and labels (i.e., news article is either fake or real). The labeling information was originally obtained from PolitiFact and Gossip Cop[3].

In this paper, our contributions are as follows:

- We hand selected and extracted three categories of features from each news article, focusing on only text information.
- We developed machine learning models based on the proposed features. In addition, we proposed and built a LSTM-attention based neural network model.
- Our models outperformed 7 baselines in FakeNewsNet [5].

II. Related Work

We briefly summarize fake news related work. Over years, researchers devoted time and effort in fake news and fact-checking domain. For example, Wang [9], Mitra et al. [10], Santia and Williams [11], Shu et al. [5] provided a wide range of publicly available datasets for better analysis of fake news. In this paper, we used a dataset collected by Shu et al. [5] and compared our models against their models as the baselines.

*The work was done when the authors were at Worcester Polytechnic Institute.
[1]https://www.snopes.com/
[2]https://www.politifact.com/
[3]https://www.gossipcop.com/

978-1-7281-0858-2/19/$31.00 ©2019 IEEE

TABLE II
TOP 10 FEATURES.

Rank	PolitiFact		Gossip Cop	
	Feature	Chi-squared Score	Feature	Chi-squared Score
1	NMF 2nd dim.	12.8	NMF 6th dim.	148.7
2	NMF 5th dim.	12.2	NMF 88th dim.	73.9
3	NMF 1st dim.	10.6	NMF 3th dim.	44.7
4	NMF 3rd dim.	10.2	NMF 42th dim.	39.2
5	|bigrams| in appeared in both title and body	9.1	NMF 10th dim.	36.1
6	|trigrams| in appeared in both title and body	9.1	NMF 16th dim.	30.7
7	NMF 4th dim.	9.0	NMF 58th dim.	19.7
8	NMF 22nd dim.	7.8	NMF 60th dim.	19.5
9	NMF 49th dim.	5.6	NMF 30th dim.	19.2
10	title's sentiment (neutral)	4.9	|sentences| in title	18.2

Researchers [1]–[3] defined fake news problems and concepts, and analyzed characteristics of fake news. Researchers in psychology worked on psychological foundations of fake news [12], [13]. Zhou and Zafarani [14] comprehensively reviewed various approach of fake news detection. Shao et al. [15] analyzed how manipulators leveraged social bots to spread fake news on social media. Shu et al. [16], [17] investigated how social context and spatial temporal data could help detect fake news. Researchers [18]–[23] utilized deep learning techniques to automatically learn feature representation and detect fake news. Other researchers also worked on detection the fake news based on traditional machine learning approaches. [24], [25]

III. DATASET AND OUR FRAMEWORK

A. Dataset

Among publicly available fake news datasets, we chose *FakeNewsNet [5]* dataset as our main dataset which contains the most comprehensive information like news content, social context, and spatiotemporal information. Since we focus on building machine learning models given only news articles, we specifically describe news content in the dataset. After doing preprocessing (which will be described in the following subsection), Table I presents statistics of news articles that we used for the rest of this paper. The ground truth/labels were obtained from two fact-checking websites, PolitiFact and Gossip Cop. PolitiFact focuses on fact-checking the U.S political news while Gossip Cop focuses on fact-checking the Hollywood and celebrity news. In the rest of paper, we will simply call the political news articles and celebrity news articles with labels as PolitiFact and Gossip Cop datasets. Interestingly, 42.3% news articles verified by PolitiFact were fake while 23% news articles verified by Gossip Cop were fake. Overall, our dataset contains 5,222 fake news articles (23.8%) and 16,729 true news articles (76.2%).

B. Our Framework

Our framework consists of preprocessing, feature extraction, machine learning and deep learning components.

1) Preprocessing: In the preprocessing step, we removed less important information such as emojis, numbers, and stop words in the remaining news articles. Punctuation were removed for all the features, except sentiment features which require punctuation in order to acquire the sentence level sentiment intensity score.

2) Feature Extraction: Given each news article, we extracted 134 features under three categories: (1) count features; (2) sentiment analysis features; and (3) term frequency inverse document frequency (TF-IDF [24]) based non-negative matrix factorization (NMF [16]) features.

- **Count Features (26 features):** The count features consist of |unigrams|, |bigrams|, |trigrams|, |sentences| of each of an article title and an article body. They also include |unique unigrams|, |unique bigrams|, |unique trigrams|, $\frac{|unique\,unigrams|}{|unigrams|}$, $\frac{|unique\,bigrams|}{|bigrams|}$, $\frac{|unique\,trigrams|}{|trigrams|}$ of each of the title and body. In addition, they include |unigrams|, |bigrams|, |trigrams|, |unique unigrams|, |unique bigrams|, |unique trigrams| appeared in both of the title and body.
- **Sentiment Analysis Features (8 features):** To understand polarity of each of the title and body of a news article, we extracted sentiment analysis features by using VADER [26] sentiment analysis tool which produces positive, neutral, negative and compound scores.
- **TF-IDF based NMF Features (100 features):** Following the Information Retrieval technique, we first extracted TF-IDF based bag of words features from each article. Since each article would be sparse in the bag of words based vector representation, we reduced the dimension size of the vector into 100 dimensions/features by using non-negative matrix factorization (NMF) [16], [27].

To measure whether all the features have distinguishing power between fake news and true news, we conducted chi-squared test. Table II shows top 10 features based on the chi-squared scores. As you can see, TF-IDF based NMF features are highly relevant to the outcome to be predicted in both political news and celebrity news. Count features are shown to be useful in the political news while not really have much correlation to celebrity news. Title related features have considerably significant relation to the prediction class.

3) Machine Learning: The last component in our framework is building machine learning models. We had a two groups of machine learning models: (1) traditionally known machine learning algorithms which directly use our proposed features; and (2) our proposed long short term memory(LSTM) with attention based neural network model. For the traditionally known algorithms, we chose Logistic Regression, SVM [6], KNN, Random Forest, AdaBoost and XGBoost [28] implemented by scikit-learn[4]. All of them directly used our 134 features as input.

4) Deep Learning: Recently deep learning models have shown their potential in the natural language processing field, especially, recurrent neural network based models produced good results for sequential data like text/news article. Therefore, we propose a long short term memory [7] with attention mechanism [8], [29] (LSTM-ATT) model to see how it performs against baselines and our other machine learning models. The LSTM-ATT directly take word embedding vectors instead of hand selected features. Thus, we first convert each word in a news article into a vector representation by using Word2Vec [30], which is one of the most common and effective ways to transform the word into a vector form. In particular, the Google's pretrained model[5] is used to produce the word embedding as this model was trained with Google News. We feed news articles into the Word2Vec neural network word by word. A 300 dimension vector is produced by the model for each word. These vectors become the input of our two layer LSTM. Each vector goes through the LSTM cell and produces an output for the next cell until all vectors are exhausted.

In the LSTM model, the first layer gets the previous input vectors and decides whether to keep the information or throw away. It is done by a "forget gate". A sigmoid layer σ takes the previous hidden state $h_{(t-1)}$ and the input x_t to decide whether retain or forget the information.

$$f_t = \sigma(W_{if}x_t + b_{if} + W_{hf}h_{(t-1)} + b_{hf}) \tag{1}$$

Next, another sigmoid layer called "input gate" i_t controls which value will be updated to the current cell, and the tanh layer g_t creates the vectors to update the current state.

$$i_t = \sigma(W_{ii}x_t + b_{ii} + W_{hi}h_{(t-1)} + b_{hi}) \tag{2}$$

$$g_t = \tanh(W_{ig}x_t + b_{ig} + W_{hg}h_{(t-1)} + b_{hg}) \tag{3}$$

Now, we could use the f_t, i_t, and g_t to calculate the current cell state c_t. Then, we perform an element-wise multiplication on the f_t and the previous cell state $c_t{-}1$ to forget the information we want to forget earlier. The product of i_t and g_t gives us new information need to update for the current cell. Adding up these two products, we get the c_t – the current cell state.

$$c_t = f_t * c_{(t-1)} + i_t * g_t \tag{4}$$

At this stage, we have everything ready to produce an output. The cell state c_t will first go though the tanh activation function and do an element-wise multiplication with the sigmoid function o_t that decides what cell state it should output. The h_t will be our final output and the whole process is repeated for the next LSTM cell.

$$o_t = \sigma(W_{io}x_t + b_{io} + W_{ho}h_{(t-1)} + b_{ho}) \tag{5}$$

$$h_t = o_t * \tanh(c_t) \tag{6}$$

We now have a collection of LSTM cells outputs. The outputs will first go though a soft attention [8] layer as shown in the Figure 1 to focus on important content. In detail, there are three steps in this attention layer to produce the context vector. The first step is to calculate the alignment score.

$$e_{ij} = v_a^\top \tanh\left(W_a s_{i-1} + U_a h_j\right) \tag{7}$$

where $W_a \in \mathbb{R}^{n\times n}$, $U_a \in \mathbb{R}^{n\times 2n}$ and $v_a \in \mathbb{R}^n$ are the weight matrices. We apply the softmax function to attention parameter e_{ij} to produce each value that lies between 0 and 1.

$$\alpha_{ij} = \frac{\exp\left(e_{ij}\right)}{\sum_{k=1}^{T_x}\exp\left(e_{ik}\right)} \tag{8}$$

Lastly, we use the attention weights α_{ij} to produce the output context vectors c_i.

$$c_i = \sum_{j=1}^{T_x} \alpha_{ij} y_i \tag{9}$$

In the figure, the soft attention layer is concatenated with mean and max of LSTM output. In addition to that, we also concatenate the previously extracted 134 hand selected features. At the end, three dropout and ReLu activation function would prevent potential over-fitting.

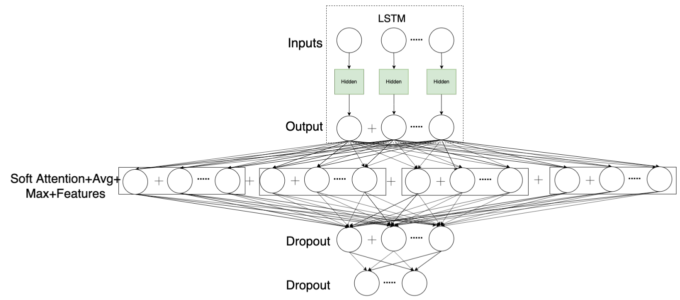

Fig. 1. Our proposed LSTM with attention model.

IV. Experiment

As we mentioned in Section III, we collected political news verified by PolitiFact, and celebrity news verified by Gossip Cop. Then, we split each dataset into training (64%), validation (16%) and test (20%) sets. We built Logistic Regression (LR), K-Nearest Neighbors (KNN), Support Vector Machines (SVM), Random Forest (RF), Adaptive Boosting (AdaBoost), and Distributed Gradient Boosting (XGBoost) based on our 134 features. In addition, we also built our proposed LSTM+ATT model. We compared our models against 7 baselines reported in the previous work [5]. To measure performance, we measured accuracy, weighted average precision, weighted average recall, and weighted average F1.

[4]https://scikit-learn.org/stable/

[5]https://code.google.com/archive/p/word2vec/

TABLE III
PERFORMANCE COMPARISON BETWEEN 7 BASELINES AND OUR MODELS WITH DEFAULT HYPERPARAMETER VALUES.

	Model	PolitiFact				Gossip Cop			
		Acc.	Pre.	Rec.	F1	Acc.	Pre.	Rec.	F1
Baselines	Logistic Regression	0.642	0.757	0.543	0.633	0.648	0.675	0.619	0.646
	SVM	0.580	0.611	0.717	0.659	0.497	0.511	0.713	0.595
	Naive Bayes	0.617	0.674	0.630	0.651	0.624	0.631	0.669	0.649
	CNN	0.629	0.807	0.456	0.583	0.723	0.751	0.701	0.725
	Social Article Fusion /S	0.654	0.600	0.789	0.681	0.689	0.671	0.738	0.703
	Social Article Fusion /A	0.667	0.667	0.579	0.619	0.635	0.589	0.882	0.706
	Social Article Fusion	0.691	0.638	0.789	0.706	0.689	0.656	0.792	0.717
Our Models	Logistic Regression	0.771	0.776	0.771	0.772	0.722	0.778	0.722	0.739
	SVM	0.598	0.357	0.598	0.447	0.318	0.746	0.318	0.257
	KNN	0.810	0.816	0.810	0.811	0.802	0.784	0.802	0.771
	Random Forest	0.821	0.822	0.821	0.822	0.819	0.805	0.819	0.803
	AdaBoost	0.788	0.786	0.788	0.785	0.804	0.790	0.804	0.794
	XGBoost	**0.832**	**0.836**	**0.832**	**0.829**	**0.842**	**0.839**	**0.842**	**0.821**
	LSTM-ATT	0.820	0.835	0.820	0.816	0.793	0.805	0.793	0.798

TABLE IV
PERFORMANCE OF OUR MODELS WITH HYPERPARAMETER TUNING.

Model	PolitiFact				Gossip Cop				PolitiFact + Gossip Cop			
	Acc.	Pre.	Rec.	F1	Acc.	Pre.	Rec.	F1	Acc.	Pre.	Rec.	F1
Logistic Regression	0.832	0.833	0.832	0.833	0.792	0.816	0.792	0.800	0.805	0.813	0.805	0.808
SVM	0.777	0.790	0.777	0.779	0.765	0.663	0.585	0.765	0.761	0.658	0.579	0.761
KNN	0.821	0.832	0.821	0.823	0.811	0.799	0.811	0.780	0.817	0.809	0.817	0.789
Random Forest	0.832	0.833	0.832	0.833	0.847	0.846	0.847	0.829	0.847	**0.854**	0.847	0.825
AdaBoost	0.821	0.821	0.821	0.821	0.838	0.829	0.838	0.827	0.824	0.812	0.824	0.812
XGBoost	**0.855**	**0.855**	**0.855**	**0.855**	**0.866**	**0.861**	**0.866**	**0.857**	**0.854**	0.849	**0.854**	**0.843**
LSTM-ATT	0.833	0.844	0.833	0.836	0.793	0.805	0.793	0.798	0.810	0.801	0.810	0.803

A. *Performance without hyperparameter tuning*

Since the prior work [5] built 7 baselines with default hyperparameter values, we report performance of our models without hyperparameter tuning for the fair comparison (i.e., only using training set to build a model without using the validation to tune hyperparmeters). Table III shows performance of the baselines and our models with default hyperparameter values in scikit-learn. Our logistic regression achieved 77.1% accuracy while the prior work's logistic regression achieved 64.2% accuracy in the PolitiFact dataset, indicating our features have more distinguishing power between fake news and true news than features used in the prior work [5]. Our XGBoost model outperformed the other models, achieving 83.2% accuracy in PolitiFact dataset and 84.2% accuracy in Gossip Cop dataset, and improving 20% in PolitiFact dataset and 16% in Gossip Cop dataset compared with the best baseline model. Our LSTM-ATT also achieved competitive results in the PolitiFact dataset, but performed worse than our XGBoost model in the Gossip Cop dataset.

B. *Performance with hyperparameter tuning*

So far we observed that our models outperformed the baselines. To even further improve the performance of our models, we tune each model's hyperparameters by using the validation set. Table IV shows performance of each of our fine-tuned models. Our XGBoost model outperformed among all of our models, achieving 85.5% accuracy and 86.6% accuracy which are higher than the previous results without fine-tuning (83.2% accuracy and 84.2% accuracy) in the PolitiFact and Gossip Cop datasets, respectively. To demonstrate that the generalization of our models, we combined the PolitiFact and Gossip Cop datasets into one dataset called PolitiFact + Gossip Cop dataset. Our models consistently performed well compared with performance in the two smaller datasets. It means our models would still perform well in more complex and cross domain news article dataset. Although LSTM-ATT performed less than XGBoost model, there would be potential to further improve its performance if we load all of the words in each article. In this study, we only loaded the first 1,000 words to make the model light.

V. CONCLUSION

In this paper, we proposed a framework to automatically detect fake news articles. In particular, we extracted 134 hand selected features from each article. Our experiments showed that the XGBoost model outperformed the best baseline model, improving 16.4% in PolitiFact dataset and 13.1% in Gossip Cop dataset compared with the best baseline. Our LSTM-ATT model achieved competitive performance in the PolitiFact dataset. With the fine tuning, the XGBoost model achieved 85.5% accuracy and 86.6% accuracy in the PolitiFact and Gossip Cop datasets, respectively. In this study, we only focused on using text information in each news article. In the future, we will incorporate additional information from social media content and spatiotemporal data for higher accuracy and early detection.

ACKNOWLEDGMENT

This work was supported in part by NSF grant CNS-1755536, AWS Cloud Credits for Research, and Google Cloud. Any opinions, findings and conclusions or recommendations expressed in this material are the author(s) and do not necessarily reflect those of the sponsors.

REFERENCES

[1] A. Vlachos and S. Riedel, "Fact checking: Task definition and dataset construction," in *Proceedings of the ACL 2014 Workshop on Language Technologies and Computational Social Science*, 2014.

[2] K. Shu, A. Sliva, S. Wang, J. Tang, and H. Liu, "Fake news detection on social media: A data mining perspective," *ACM SIGKDD Explorations Newsletter*, vol. 19, no. 1, pp. 22–36, 2017.

[3] H. Allcott and M. Gentzkow, "Social media and fake news in the 2016 election," *Journal of economic perspectives*, vol. 31, no. 2, pp. 211–36, 2017.

[4] R. Lab, "Global fact-checking up 50% in past year." http://reporterslab.org/global-fact-checking-up-50-percent, 2016.

[5] K. Shu, D. Mahudeswaran, S. Wang, D. Lee, and H. Liu, "Fakenewsnet: A data repository with news content, social context and dynamic information for studying fake news on social media," *arXiv preprint arXiv:1809.01286*, 2018.

[6] K. Crammer and Y. Singer, "On the algorithmic implementation of multiclass kernel-based vector machines," *J. Mach. Learn. Res.*, vol. 2, pp. 265–292, Mar. 2002.

[7] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, pp. 1735–1780, Nov. 1997.

[8] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," *CoRR*, vol. abs/1409.0473, 2014.

[9] W. Y. Wang, "liar, liar pants on fire: A new benchmark dataset for fake news detection," *arXiv preprint arXiv:1705.00648*, 2017.

[10] T. Mitra and E. Gilbert, "Credbank: A large-scale social media corpus with associated credibility annotations," in *ICWSM*, 2015.

[11] G. C. Santia and J. R. Williams, "Buzzface: A news veracity dataset with facebook user commentary and egos," in *ICWSM*, 2018.

[12] R. S. Nickerson, "Confirmation bias: A ubiquitous phenomenon in many guises," *Review of General Psychology*, vol. 2, no. 2, pp. 175–220, 1998.

[13] A. Ward, "Naive realism in everyday life: Implications for social conflict and misunderstanding," *Values and knowledge*, pp. 103–135.

[14] X. Zhou and R. Zafarani, "Fake news: A survey of research, detection methods, and opportunities," 2018.

[15] C. Shao, G. L. Ciampaglia, O. Varol, A. Flammini, and F. Menczer, "The spread of fake news by social bots," *arXiv preprint arXiv:1707.07592*, 2017.

[16] K. Shu, S. Wang, and H. Liu, "Beyond news contents: The role of social context for fake news detection," in *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*, WSDM '19, (New York, NY, USA), pp. 312–320, ACM, 2019.

[17] K. Shu, S. Wang, and H. Liu, "Exploiting tri-relationship for fake news detection," *arXiv preprint arXiv:1712.07709*, 2017.

[18] N. Ruchansky, S. Seo, and Y. Liu, "Csi," *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management - CIKM '17*, 2017.

[19] Y. P. Liu and Y. fang Brook Wu, "Early detection of fake news on social media through propagation path classification with recurrent and convolutional networks," in *AAAI*, 2018.

[20] Y. Wang, F. Ma, Z. Jin, Y. Yuan, G. Xun, K. Jha, L. Su, and J. Gao, "Eann: Event adversarial neural networks for multi-modal fake news detection," in *Proceedings of the 24th acm sigkdd international conference on knowledge discovery & data mining*, pp. 849–857, ACM, 2018.

[21] Y. Yang, L. Zheng, J. Zhang, Q. Cui, Z. Li, and P. S. Yu, "Ti-cnn: Convolutional neural networks for fake news detection," 2018.

[22] J. Zhang, B. Dong, and P. S. Yu, "Fakedetector: Effective fake news detection with deep diffusive neural network," 2018.

[23] F. Monti, F. Frasca, D. Eynard, D. Mannion, and M. M. Bronstein, "Fake news detection on social media using geometric deep learning," *ArXiv*, vol. abs/1902.06673, 2019.

[24] V. Pérez-Rosas, B. Kleinberg, A. Lefevre, and R. Mihalcea, "Automatic detection of fake news," in *Proceedings of the 27th International Conference on Computational Linguistics*, (Santa Fe, New Mexico, USA), pp. 3391–3401, Association for Computational Linguistics, Aug. 2018.

[25] X. Zhou, A. Jain, V. V. Phoha, and R. Zafarani, "Fake news early detection: A theory-driven model," 2019.

[26] C. J. Hutto and E. Gilbert, "Vader: A parsimonious rule-based model for sentiment analysis of social media text," in *ICWSM*, 2014.

[27] W. Xu, X. Liu, and Y. Gong, "Document clustering based on non-negative matrix factorization," in *SIGIR*, 2003.

[28] T. Chen and C. Guestrin, "Xgboost," *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining - KDD '16*, 2016.

[29] J. Cheng, L. Dong, and M. Lapata, "Long short-term memory-networks for machine reading," in *EMNLP*, 2016.

[30] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," 2013.