Creating Stochastic Text Data to Solve Privacy Issues in Social Networking

Abdolreza Abhari
Department of Computer Science
Ryerson University
Toronto, Canada
aabhari@scs.ryerson.ca

Jason Li
Department of Computer Science
Ryerson University
Toronto, Canada
jcli@ryerson.ca

Nicholas Buhagiar
Department of Computer Science
Ryerson University
Toronto, Canada
nbuhagiar@ryerson.ca

***Abstract*—This work addresses one of the privacy concerns regarding testing social media applications, where testing applications such as recommender systems requires input data from real users. The aim of this paper is to show how probability distributions, in particular the Weibull model, can be used to generate large collections of artificial unstructured data to simulate real data for testing recommender systems. In this work we analyzed data from two social media websites: user tweets on Twitter and comments made in discussion forums on Reddit, both of which contain several thousand data instances.**

Keywords— Social Networks, Curve fitting, Recommender Systems

I. INTRODUCTION

When developing social network software applications such as recommender systems, testing performance requires large amounts of user data. This user data is usually in the form of text such as tweets in Twitter and posts on Facebook. Presently, collecting this real data is sometimes done by hiring real users, such as in the cases of [1, 2]. The issue with this method is that it limits the amount of data one has available and is not feasible when developing massive data mining applications such as recommender systems. To circumvent this issue, large public or private social media related datasets can be accessed, though this sometimes requires special permission to use. However, these users whose social media activity is present in these datasets may express concern about how their data is being analyzed and for what purposes. Therefore, user privacy is one of the major concerns when measuring performance on social network applications that require huge amounts of data to be tested. Generating large amounts of artificially generated data is a promising solution to this problem, and how the results of using this artificial data relates to those achieved by real data will be discussed in this work. In this work, we collected real data from Twitter users and also created large amounts of stochastic data to represent users' tweets. We then developed a recommender system for these two datasets and compared their results. We also examined our model for generating stochastic tweets by using it in conjunction with comments made by users in different discussion threads within a discussion forum.

II. RELATED WORK

Generating large amounts of artificial data has been previously proposed for database systems which deal with structured data such as tables, records, etc. SimSQL, for example, uses machine Learning Bayesian methods to generate stochastic data for database simulations [3]. The Synthetic Data Vault as well uses machine learning approaches to build generative models of relational databases [4]. In a previous paper, we also presented a method that can be used to generate stochastic Tweets [5]. To the best of our knowledge, this paper is the first work that suggests creating artificial unstructured data to circumvent privacy issues in social networking.

Though to the best of our knowledge there is no similar work related to generating these stochastic tweets, research has been done in modeling tweets, by aggregating all of the tweets made by a particular user into a document [6, 7]. However, these aggregated tweets were not used to generate stochastic data.

With regards to simulating Twitter itself, the software package Hashkat was developed as a means in which to simulate large social networks structure and analyze network topology growth as well as information flow within them using agent-based modeling [8]. In addition, [9] simulated a Twitter financial community to analyze information flow using a multi-agent system. Both simulation studies achieved results similar to what has been found to occur in the real world however the concentration of these works are simulation of network and studies the topology of subnetworks not generating artificial tweets. [8, 9].

With regards to simulating a recommender system, our group previously developed a multi-agent system to simulate a distributed recommender system in order to analyze its scalability [10]. The recommendation algorithms used in this simulation were cosine similarity and k-means clustering, with the dataset comprising of tweets made by real Twitter users. Another simulation of a recommender system using a multi-agent system was also created by [11], though this work did not focus on Twitter messages.

III. METHODOLOGY

Three datasets were used in this research. *Real_Tweets_DS* consisted of 14,000 real tweets made by followers of Ryerson University's Twitter account, "RyersonU", and "RyersonU" itself. The tweets of each user were gathered from the day before they followed RyersonU. *Generated_Tweets_DS* consisted of 14,000 generated tweets, which were generated by the model proposed in this document. *Comments_DS* consisted of 10,000 comments made by 8,282 users in 6,344 discussion threads on Reddit, and were acquired by Jason Baumgartner of pushshift.io using Reddit's API. All data sets with their counts are shown in Table 1.

TABLE 1: NUMBER OF TWEETS/COMMENTS FOR EACH DATASET

Dataset	**Number of Tweets/Comments**
Real_Tweets_DS	14,000
Generated_Tweets_DS	14,000
Comments_DS	10,000

TF·IDF is a common method of finding word importance in documents, and is used by most recommender systems. The TF·IDF score for a given word *i* in a document *j* in a corpus is the product of the word's term frequency (TF) in that document with the word's inverse document frequency (IDF) in the corpus. A word's term frequency within a document is defined as the number of its occurrences within the document, normalized by dividing it by the most frequently occurring word within that same document. This can be summarized by the following equation:

$$TF_{ij} = \frac{f_{ij}}{\max_k f_{kj}} \tag{1}$$

where f_{ij} is the count of word *i* in document *j*, and $\max_k f_{kj}$ is the count of the most frequently occurring word *k* in document *j* [12]. The inverse document frequency is a score for the number of documents in a corpus a word appears in, penalizing words that appear in a large portion of them, and can be calculated in the following manner:

$$IDF_i = \log_2\left(\frac{N}{d_i}\right) \tag{2}$$

where *N* is the number of documents in the corpus and d_i is the number of documents that word *i* appears in [12]. The TF·IDF score for word *i* in document *j* can therefore be calculated as:

$$TF \cdot IDF_{ij} = TF_{ij} \cdot IDF_i \tag{3}$$

for every word in the corpus [12].

The TF·IDF score was determined for terms in the *Real_Tweets_DS* dataset after tweets were processed and aggregated into documents where each document represented a user. Bins were created for each TF·IDF score and the words that were associated with it.

After calculating the TF·IDF score for different users, a Weibull model appeared to be the best fit for the dataset (further details can be found at [5]). This Weibull Distribution was used to generate a random number that represented a TF·IDF score, and its associated word or the word in the closest bin to it was then selected to be used in a generated artificial tweet. Tweets ranging from 3 to 7 words were generated, ignoring stop words. The scale and shape parameters of the Weibull distribution used for the *Real_Tweets_DS* dataset are shown in Table 2. The average of these parameters were used to generate artificial data.

TABLE 2: WEIBULL DISTRIBUTION SCALE AND SHAPE PARAMETERS FOR REAL_TWEETS_DS

User	Scale	Shape
B	2.26E-08	0.29282
C	6.51E-07	0.17736
D	1.46E-05	0.16122
E	1.48E-06	0.16538
F	3.21E-06	0.16212
G	6.53E-07	0.17806
H	8.43E-07	0.1718
I	9.00E-07	0.16484
J	2.98E-07	0.18279
K	1.69E-06	0.15725
L	2.13E-08	0.30634
M	1.95E-06	0.1599
Average	2.19E-06	0.18999

Fig. 1-4 shows a comparison of a fitted Weibull distribution with empirical data from real tweets for four users, which was found to be representative of the results obtained for the rest of the Twitter users in our dataset.

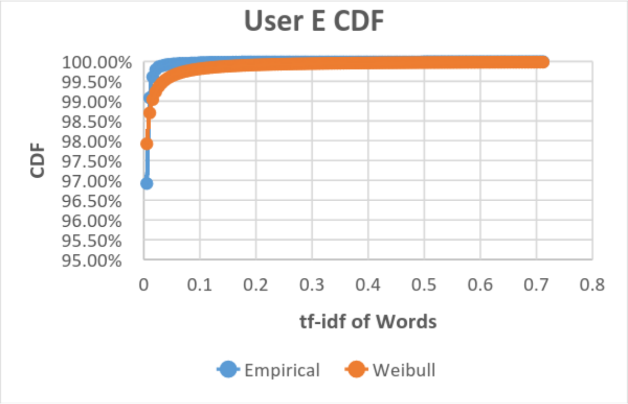

Fig. 1. CDF of User E.

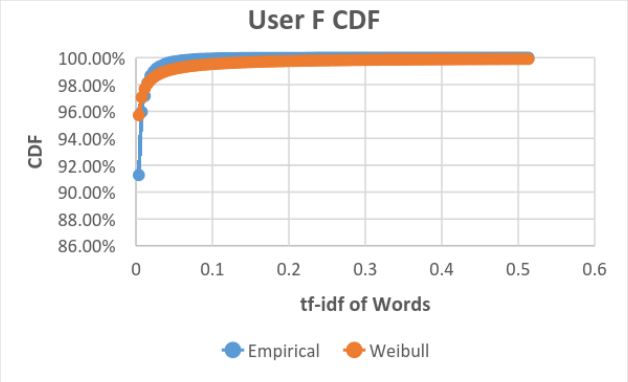

Fig. 2. CDF of User F.

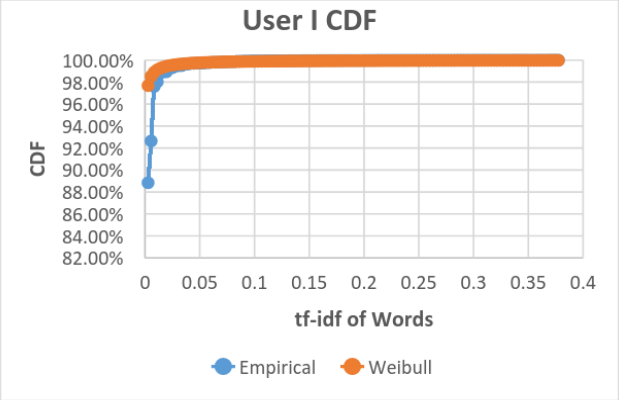

Fig. 3. CDF of User I.

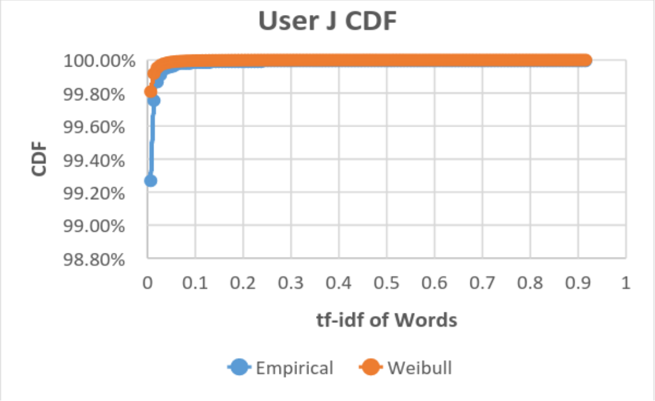

Fig. 4. CDF of User J.

For *Comments_DS*, a Latent Dirichlet Allocation (LDA) topic model [13] was first used to construct documents out of the words present in the dataset. To make this dataset suitable for training an LDA model, these comments were tokenized and lemmatized using the *Natural Language ToolKit* (*NLTK*) library in Python [14], with stop words appearing in Python's English stop words library [15] being removed as well as any single character tokens. Vectorizing these collections of tokens using the Python machine learning package *scikit-learn* [16], an LDA model consisting of 5 topics was then trained on this dataset using the Python topic modeling package *gensim* [17]. Once training of this LDA model was completed, the 100 most frequently occurring tokens in each topic were extracted, as well as their corresponding probability of appearing in that topic. Treating these topics as documents and these term weights as counts, the term frequency for each token in a given document was calculated by dividing these probabilities by the probability of the most probable word occurring for that topic. The inverse document frequency was also determined for each token that appeared in any of these five topics.

Using these scores to obtain the TF·IDF score for each word in each topic, we determined the scale and shape parameters of the Weibull distribution for each topic, as can be seen in Table 3. As demonstrated in Fig. 5, Fig. 6, Fig. 7, Fig. 8, and Fig. 9, the Weibull distribution of this dataset can again be seen as a good fit for empirical data. We used ExpertFit which is a curve fitting software to test Weibull model for all mentioned data sets and another text-based conversation like data sets and Weibull distribution was found a good fit when using TF-IDF in processing text data.

TABLE 3: WEIBULL DISTRIBUTION SCALE AND SHAPE PARAMETERS FOR CONVERSATIONDS

User	Scale	Shape
Topic 0	7.63E-05	0.12604
Topic 1	6.59E-05	0.12832
Topic 2	7.72E-05	0.12618
Topic 3	5.21E-05	0.13202
Topic 4	7.15E-05	0.12718

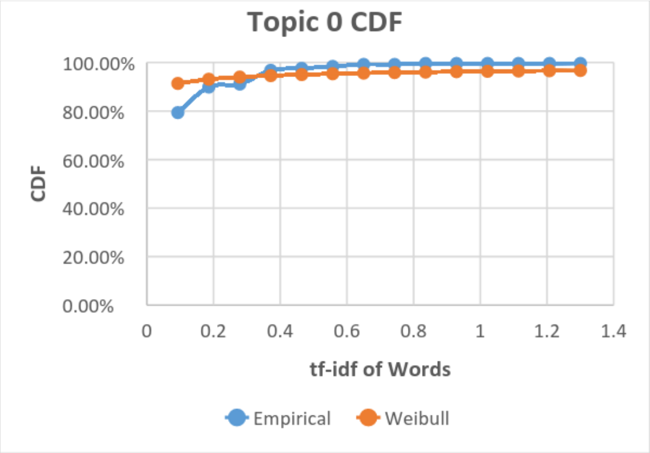

Fig. 5. CDF of Topic 0.

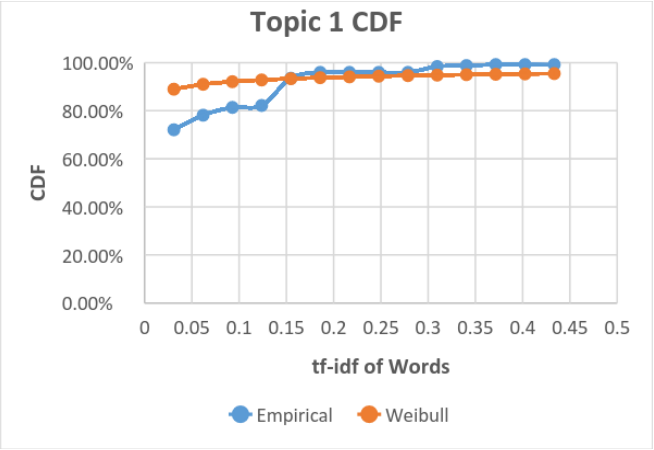

Fig. 6. CDF of Topic 1.

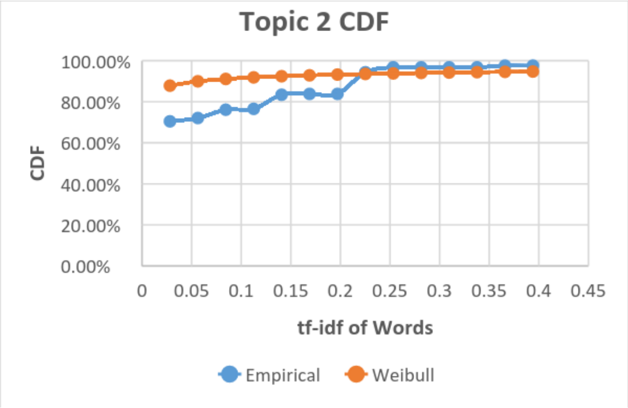

Fig. 7. CDF of Topic 2.

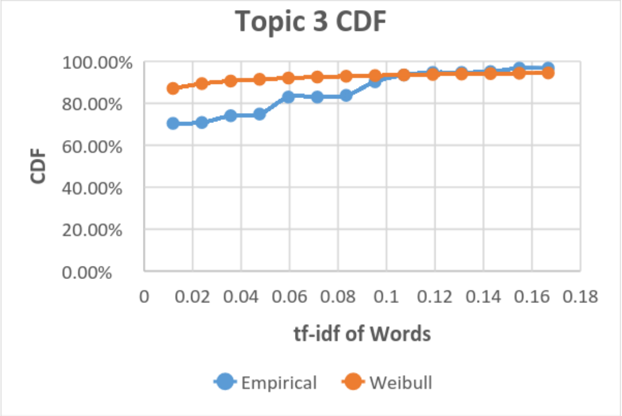

Fig. 8. CDF of Topic 3.

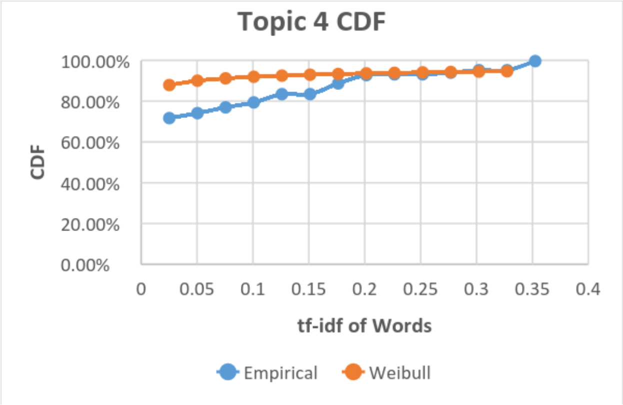

Fig. 9. CDF of Topic 4.

IV. EXPERIMENTAL RESULTS

To compare the results obtained by our generated tweets with real data, we simulated a recommender system based on k-means Clustering for both the *Real_Tweets_DS* and *Generated_Tweets_DS* datasets.

A value of 3 was used for *k*, and cosine similarity was used as a distance metric due to the data instances being clustered consisting of document vectors [18]. The average value assigned to each term in the document vectors are the weights (i.e. TF-IDF) for the centroids in the clusters [19].

A node in a distributed environment is an instance that can be a cluster or perhaps another server with its local chunk of data (e.g. a node in a Hadoop file system consists of a mapper and a reducer) The main benefits of generating a large dataset of stochastic tweets for a recommender system is that we can test its performance using parallel implementations of the recommendation algorithm when increasing the number of nodes (further detail can be found in [5]). In this experiment, recommender systems with 1, 2, and 4 configurations of nodes were analyzed, with the k-means recommendation algorithm generating recommendations for 10 users. This was done for both the *Real_Tweets_DS* and *Generated_Tweets_DS* datasets.

The timing and communication cost was measured for all these instances, as well as the recommendation accuracy, where recommendation accuracy was determined by whether or not "RyersonU" was in the Top-10 Twitter accounts recommended for a user to follow. The results of this analysis can be seen in Fig. 10-13.

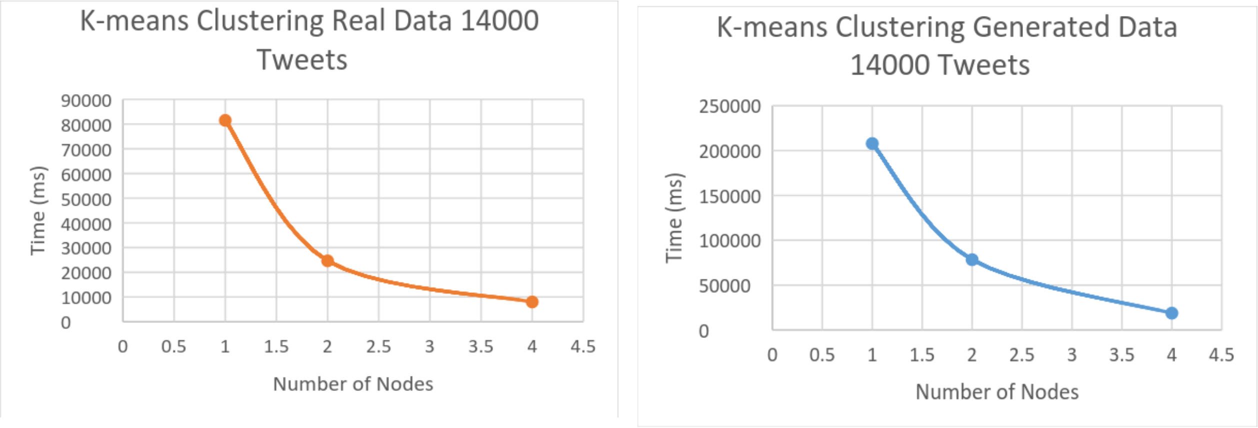

Fig. 10. Timing of k-means algorithm on TweetsDS A and B.

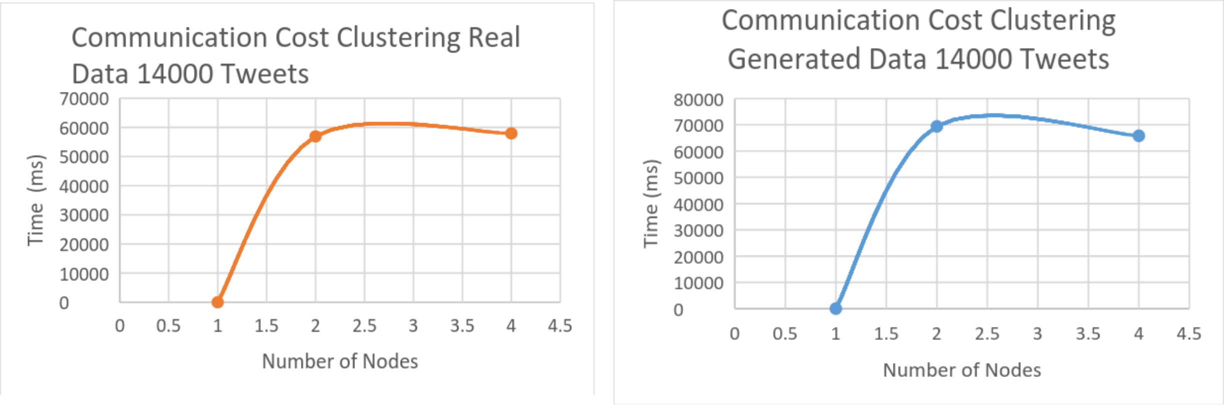

Fig. 11. Communication cost for TweetsDS A and B.

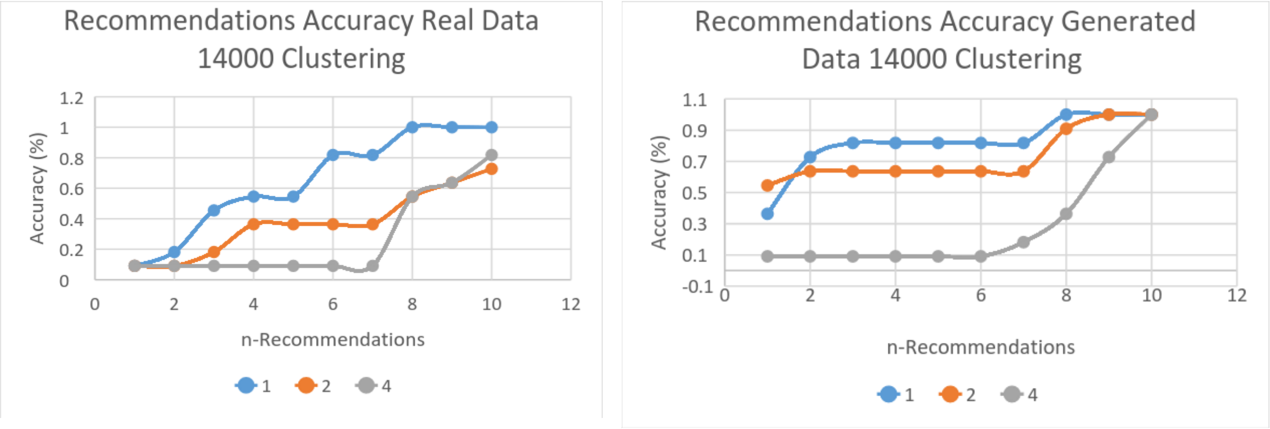

Fig. 12. Recommendation accuracy for TweetsDS A and B.

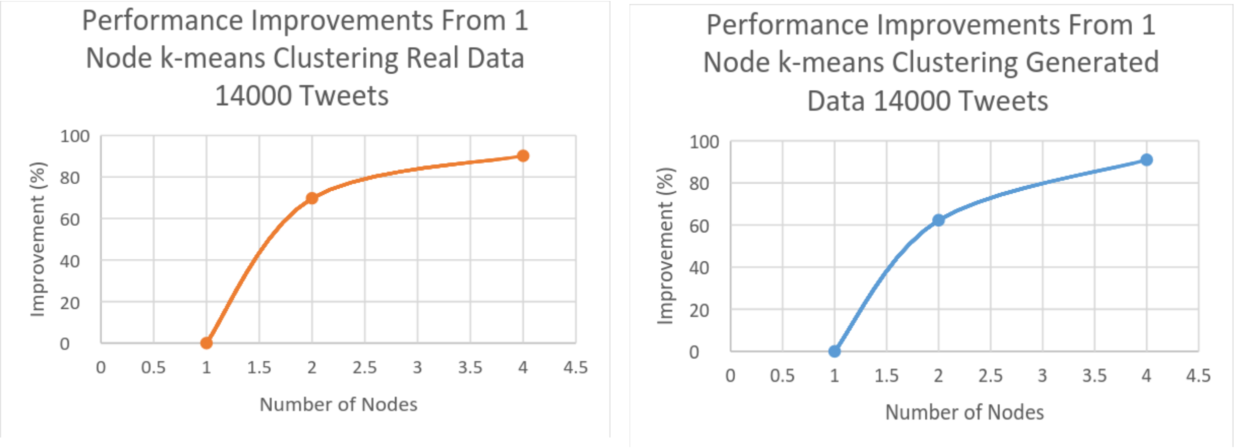

Fig. 13. Performance Improvements for TweetsDS A.

In Fig. 13, one can see the performance improvements for these two datasets when increasing the number of nodes. Since the results are similar, we can thus conclude that our proposed method for generating artificial tweets to test the performance of recommender systems provides similar results to real data.

V. CONCLUSION

Many researchers are in constant need to simulate social network applications and test their performance using massive data mining algorithms. The benefits of using parallelization in these approaches are also something that needs to be considered. However, due to the privacy issue and the time-consuming nature of collecting real data, this task is not easy. In this work, the Weibull distribution proved to be a useful model for generating stochastic data to simulate real text data when selecting the words with TF-IDF method. We used generated tweets produced by Weibull model for testing a recommender system's performance. Weibull models are also shown to be appropriate for modeling other social media datasets such as comments made in a discussion forum. Testing our recommender system with both real and generated tweets, we were able to find that artificial data provided similar results to real data. Thus, by using stochastic data instead of real data for testing social media software application performance, the privacy issue can be avoided.

Being able to create artificial data makes it possible to scale unstructured data in the millions, enabling more tests to be done with massive data mining applications. Future work for this research will focus on the testing of recommender systems on this unstructured data from other social media platforms (e.g. posts on Facebook, comments from Reddit, etc.).

ACKNOWLEDGMENT

Thanks must be extended to Jason Baumgartner of pushshift.io for taking the time to collect the Reddit comment data we studied using Reddit's own API. His time and effort are greatly appreciated it.

REFERENCES

[1] R. A. C. Capuruço and L. F. Capretz, "Evaluation and Assessment of Recommenders Using Monte Carlo Simulation," presented at the *SRS 2012: Social Recommender Systems, Workshop and Poster Proc. 20th Conference User Modeling, Adaptation, and Personalization, CEUR Workshop*, Montreal, Canada, 2012.

[2] R. A. C. Capuruço and L. F. Capretz, "A fuzzy-based inference mechanism of trust for improved social recommenders," presented at the *SRS 2012: Social Recommender Systems, Workshop and Poster Proc. 20th Conference User Modeling, Adaptation, and Personalization, CEUR Workshop*, Montreal, Canada, 2012.

C. Jermaine et al. *SimSQL* [Online]. Available: http://cmj4.web.rice.edu/SimSQL/SimSQL.html

[3] [12] J. Leskovec, A. Rajaraman, and J. D. Ullman, *Mining of Massive Datasets*. Cambridge, United Kingdom: Cambridge University Press, 2014.

[4] N. Patki, R. Wedge, and K. Veeramachaneni, "The Synthetic Data Vault," in *2016 IEEE International Conference Data Science and Advanced Analytics (DSAA)*, Montreal, Canada, 2016, pp. 399-410.

[5] J. Li and A. Abhari, "Generating stochastic data to simulate a twitter user," presented at the *Proc. 20th Communications & Networking Symposium*, Virginia Beach, VA, 2017.

[6] D. Alvarez-Melis and M. Saveski, "Topic Modeling in Twitter: Aggregating Tweets by Conversations," in *Proc. Tenth International AAAI Conference Web and Social Media (ICWSM 2016)*, Palo Alto, CA, 2016, pp. 519-522.

[7] L. Hong and B. D. Davison, "Empirical Study of Topic Modeling in Twitter," in *Proc. First Workshop Social Media Analytics*, Washington D.C., 2010, pp. 80-88.

[8] K. Ryczko, A. Domurad, N. Buhagiar, and I. Tamblyn, "Hashkat: large-scale simulations of online social networks," in *Social Network Analysis and Mining*, no. 1, Vienna, Austria: Springer, 2017, pp. 4.

[9] S. Y. Yang, A. Liu, and S. Y. K. Mo, "Twitter Financial Community Modeling using Agent Based Simulation," in *Proc. 2014 IEEE Conference Computational Intelligence for Financial Engineering & Economics (CIFEr)*, Piscataway, NJ, 2014, pp. 55-62.

[10] L. Ahmed, "Distributed Recommender System Using Multi-Agent For Social Networks," Ph.D. dissertation, Dept. Com. Sci., Ryerson Univ., Toronto, Canada, 2016.

[11] R. Saga, K. Okamoto, H. Tsuji, and K. Matsumoto, "Proposal of a recommender system simulator based on a small-world model," in *Artificial Life and Robotics*, vol. 16, Japan: Springer, 2011, pp. 426-429.

[12] J. Leskovec, A. Rajaraman, and J. D. Ullman, *Mining of Massive Datasets*. Cambridge, United Kingdom: Cambridge University Press, 2014.

[13] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent dirichlet allocation," in *Journal of machine learning research*," vol. 3, no. Jan, 2003, pp. 993-1022.

[14] S. Bird, E. Klein, and E. Loper, *Natural language processing with Python: analyzing text with the natural language toolkit*," O'Reilly Media, Inc., 2009.

[15] A. Savand. (2015, February 23). *Stop-words 2015.2.23.1: Python Package Index* [Online]. Available: https://pypi.python.org/pypi/stop-words

[16] F. Pedregosa et al., “Scikit-learn: machine learning in Python,” in *Journal of Machine Learning Research*, vol. 12, no. Oct, 2011, pp. 2825-2830.

[17] R. Rehurek and P. Sojka, “Software framework for topic modelling with large corpora,” in *Proc. LREC 2010 Workshop New Challenges for NLP Frameworks*, Valetta, Malta, 2010, pp. 46-50.

[18] V. K. Singh, N. Tiwari, and S. Garg, “Document Clustering using K-means, Heuristic K-means and Fuzzy C-means,” in *Proc. 2011 International Conference Computational Intelligence and Communication Systems*, Piscataway, NJ, 2011, pp. 297-301.

[19] E. H. S. Han and G. Karypis, “Centroid-Based Document Classification: Analysis & Experimental Result,” in *Proc. 4th European Conference Principles and Practice of Knowledge Discovery in Databases (PKDD)*, Lyon, France, 2000, pp. 424-431.