LSHWE: Improving Similarity-Based Word Embedding with Locality Sensitive Hashing for Cyberbullying Detection

Zehua Zhao*, Min Gao*, Fengji Luo†, Yi Zhang* and Qingyu Xiong*
*School of Big Data and Software Engineering
Chongqing University, Chongqing, China
zhzhao@cqu.edu.cn, gaomin@cqu.edu.cn, cquzhangyi@cqu.edu.cn, xiong03@cqu.edu.cn
†School of Civil Engineering
The University of Sydney, Sydney, Australia
fengji.luo@sydney.edu.au

***Abstract*—Word embedding methods use low-dimensional vectors to represent words in the corpus. Such low-dimensional vectors can capture lexical semantics and greatly improve the cyberbullying detection performance. However, existing word embedding methods have a major limitation in cyberbullying detection task: they cannot represent well on "deliberately obfuscated words", which are used by users to replace bullying words in order to evade detection. These deliberately obfuscated words are often regarded as "rare words" with a little contextual information and are removed during preprocessing. In this paper, we propose a word embedding method called LSHWE to solve this limitation, which is based on an idea that deliberately obfuscated words have a high context similarity with their corresponding bullying words. LSHWE has two steps: firstly, it generates the nearest neighbor matrix according to the co-occurrence matrix and the nearest neighbor list obtained by Locality Sensitive Hashing (LSH); secondly, it uses an LSH-based autoencoder to learn word representations based on these two matrices. Especially, the reconstructed nearest neighbor matrix generated by the LSH-based autoencoder is used to make the representations of deliberately obfuscated words close to their corresponding bullying words. In order to improve the algorithm efficiency, LSHWE uses LSH to generate the nearest neighbor list and the reconstructed nearest neighbor list. Empirical experiments prove the effectiveness of LSHWE in cyberbullying detection, particularly on the "deliberately obfuscated words" problem. Moreover, LSHWE is highly efficient, it can represent tens of thousands of words in a few minutes on a typical single machine.**

***Index Terms*—Word Embedding, Locality Sensitive Hashing, Cyberbullying Detection**

I. Introduction

While the development of the Internet has brought convenience to people, it has also brought a series of negative effects, cyberbullying is one of them. Cyberbullying refers to a kind of bullying that takes place over digital devices like cell phones, computers, and tablets. It can be launched through a variety of media, such as textual messages, online forums, electronic games, etc. [1].

Compared with traditional bullying, cyberbullying can spread more widely and rapidly by virtue of the characteristics of the network. Ditchthelabel and Habbo Hotel [2] surveyed 10,008 teenagers aged 13-22 years old and found cyberbullying had caused serious negative effect on adolescents: 37% of these teenagers had experienced cyberbullying on a frequent basis. Therefore, cyberbullying detection has become a hot topic in recent years, aiming at reducing the negative impact of cyberbullying on society, especially adolescents.

Cyberbullying detection methods can be divided into two categories, textual-based detection methods and multimodal-based detection methods. Textual-based detection methods only use textual information to detect cyberbullying events. Most of these methods are based on dictionaries, using special words (e.g. slangs and emotional keywords) as features [3]–[6]. Compared with textual-based detection methods, multimodal-based detection methods incorporate additional information (e.g. time and geographical information) to restore the environment in which the cyberbullying events occur to improve the detection performance [7]–[10].

However, whether textual-based detection methods or multimodal-based detection methods, text representation learning is the core of cyberbullying detection task. Recently, many studies use word embedding methods to learn text representations and perform cyberbullying detection based on that [11]–[13]. Word embeddings are real-valued word representations able to capture lexical semantics and trained on natural language corpora [14].

There is one major limitation in existing work. In cyberbullying events, users could artificially obfuscate some bullying words (such as "fxxk" and "fcukk" in Fig. 1) in order to evade detection. These words are called "deliberately obfuscated words" [15] and are often regarded as "rare words" which have a little contextual information. Existing word embedding methods usually remove these rare words during preprocessing, but these rare bullying words are the core of cyberbullying detection task and should not be removed.

Inspired by " 'Deliberately obfuscated words' have a high context similarity with their corresponding bullying words", we design a similarity-based word embedding method LSHWE (**L**ocality **S**ensitive **H**ashing-based **W**ord

978-1-7281-6926-2/20/$31.00 ©2020 IEEE

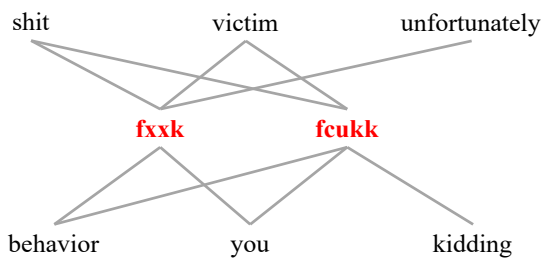

Fig. 1. Example of Deliberately Obfuscated Words.

Embedding) aiming at solving the "deliberately obfuscated words" problem in cyberbullying detection task. The representations of those "deliberately obfuscated words" learnt through LSHWE could be as close as possible to their corresponding bullying words' representations, which can effectively improve the cyberbullying detection performance. Our contributions can be summarized as follows:

- To propose a new word embedding method "LSHWE" that can effectively represent deliberately obfuscated words in cyberbullying events. The proposed method uses a Nearest Neighbor Search (NN) method to identify words which have a high context similarity with the rare words. Based on this, an autoencoder model is used to learn words' representations. Experiments show that LSHWE can alleviate the "obfuscated bullying words" problem.
- Locality Sensitive Hashing is used to search the nearest neighbors for each rare word to improve the algorithm efficiency. Experiments demonstrate that LSH-based word embedding method can greatly improve the computational efficiency of text representation learning, especially on large-scale datasets.

II. Related Work

This section gives a brief review on the researches related to this study, which can be generally categorized as below.

A. Text Representation Learning

Text representation learning is the core task in both textual-based cyberbullying detection methods and multimodal-based cyberbullying detection methods.

Bag-of-Words (BoW) model is a basic text representation model that uses a binary vector to represent each text in a corpus. The vector's dimension is the size of the corpus; each element in the vector represents one word. Although BoW model is easy to implement, it shows limitations on extracting latent features of the text; moreover, the sparsity of the binary vectors generated by BoW makes them inefficient in many applications. N-grams is another widely used text representation method [16]. It is a type of probabilistic language model and can predict the next item through a $(n-1)$-order Markov sequence. N-grams model can effectively represent the semantic association between words, but with the increase of n, it will face the "Curse of Dimensionality" problem. While both BoW and N-gram are learning representation from unstructured text, there is also attempt learning representation leveraging both unstructured text and structured data (i.e., linked data) [17], [18].

In recent years, many neural network-based word embedding methods [19] have been proposed, providing a new way for text representation learning. These methods efficiently calculate the semantic association between words and generate low-dimensional vectors to represent the words [20]. Word2vec [21] is a typical word embedding method, which has two models: Continuous Bag-of-Words (CBOW) and Skip-gram. CBOW learns word representations through predicting the target word according to its contextual words, while Skip-gram predicts a known target word's contextual words. In 2014, Pennington et al. [22] proposed another famous word embedding method "GloVe". Different from Word2vec that only uses the information obtained from a local context window, GloVe extracts the corpus's global statistic information based on the global co-occurrence matrix. Recently, more and more Transformer-based language models are proposed. For example, Bert [23] could learn words' features from both forward and backward, and experiments show that it outperforms in many tasks.

Word embedding methods have achieved great performance in many tasks, such as sequence modelling [24] and metaphor processing [25]. However, the removal of rare words in pre-processing procedure makes most of them have a limitation on solving the "deliberately obfuscated words" problem in cyberbullying detection task.

B. Nearest Neighbor Search

Nearest neighbor search is to find the items in a dataset which are similar with the target item. There are two types of nearest neighbor search methods: Exact Nearest Neighbor Search (Exact NN) methods and Approximate Nearest Neighbor Search (ANN) methods. Although Exact NN methods can get an exact search result, they are only suitable for small datasets with low dimension. For massive datasets with high dimension, these methods will have an exponential complexity.

To improve the computational efficiency of Exact NN methods, some researchers proposed the ANN methods. Generally, ANN methods can be defined as "Given a set S of n points in a d-dimensional space R^d, construct a data structure which given any query point $q \in R^d$, reports any point within distance at most c times the distance from q top, where p is the point in S closest to q" [26]. One method to construct such data structure is via oblivious dimension-reduction. It firstly performs dimension reduction on the ANN's input vectors, and then searches for the nearest neighbors in the dimension reduced vectors. While such dimension reduction yields data structures with the polynomial space complexity, it still can hardly to be applied in real-world applications, in which the solution's space complexity is desired to have a linear relationship with the number of input vectors [27]. To achieve this, some researchers proposed another type of

978-1-7281-6926-2/20/$31.00 ©2020 IEEE

ANN method based on randomized space partitions, which is to project the input vectors into a random subspace while performs dimension reduction [28].

In this paper, we use a typical randomized space partitions-based ANN method called "Locality Sensitive Hashing (LSH)" to search the nearest neighbors for each rare word.

III. LSHWE: Improving Similarity-Based Word Embedding with Locality Sensitive Hashing

In this section, we give a description of the new word embedding method that we proposed.

A. Framework of LSHWE

Given a corpus S consisting m short sentences. The corpus's size is n, i.e. the total number of words in the m short sentences. Assuming there are r rare words in the corpus, where a word is identified to be a rare word if its occurrence frequency in the corpus is less than a threshold b. LSHWE maps each word in S to a low-dimensional vector, which could subsequentially be used in cyberbullying detecting.

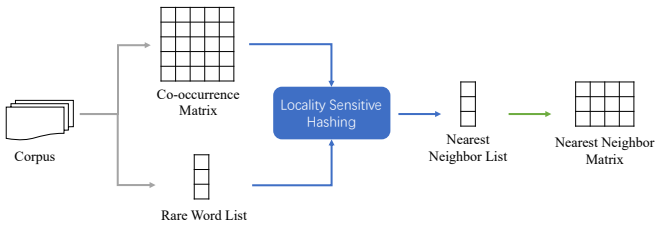

Fig. 2. Step (1) in LSHWE: Generation of Nearest Neighbor Matrix.

LSHWE has two steps: (1) it generates a nearest neighbor matrix; and (2) it uses an autoencoder model to learn word representations.

For step (1), as shown in Fig. 2, LSHWE generates a co-occurrence matrix $C \in \mathbb{R}^{n\times n}$ and a rare word list $R \in \mathbb{R}^r$ according to the given corpus S firstly. Note that C_{ij} represents the frequency of word i co-occurring with word j, and $C_i = \sum_{j=1}^{n} C_{ij}$ is the total number of occurrences of word i in corpus. Secondly, locality sensitive hashing is used to generate a nearest neighbor list $NL \in \mathbb{R}^{r\times k}$ based on C and R. Note that NL_{ij} represents the jth neighbor of rare word R_i and k is a pre-defined parameter represents the top-k nearest neighbors. Finally, a nearest neighbor matrix $N \in \mathbb{R}^{r\times k}$ is generated using the following equation:

$$N_{ij} = dist(C_{R_i}, C_{NL_{ij}}), \tag{1}$$

where $dist(a, b)$ is a distance function.

For step (2), as shown in Fig. 3, an LSH-based autoencoder model is used to learn each word's representation. The LSH-based autoencoder model contains two parts: an encoder and a decoder. LSHWE uses two two-layer full-connection neural network as encoder and decoder, respectively. The input of the encoder is the co-occurrence matrix C and the output is latent word vectors $L \in \mathbb{R}^{n\times d}$, where d is a pre-defined parameter represents the dimension of word vectors. The input

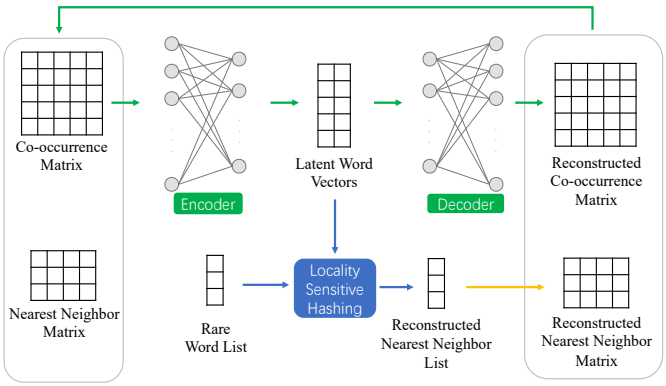

Fig. 3. Step2 in LSHWE: Autoencoder-Based Word Representation Learning.

of the decoder is the latent word vectors and the output is a reconstructed co-occurrence matrix $\hat{C} \in \mathbb{R}^{n\times n}$.

Different from the traditional autoencoder model, LSHWE generates a reconstructed nearest neighbor matrix. Firstly, for the latent word vectors L and rare word list R, locality sensitive hashing is used again to generate a reconstructed nearest neighbor list $\hat{NL} \in \mathbb{R}^{r\times k}$. Then, LSH-based autoencoder generates a reconstructed nearest neighbor matrix $\hat{N} \in \mathbb{R}^{r\times k}$ according to Eq. 2:

$$\hat{N_{ij}} = \begin{cases} dist(L_{R_i}, L_{N\hat{L}_{ij}}), & \text{if } N\hat{L}_{ij} = NL_{ij} \\ -1, & \text{otherwise} \end{cases}, \tag{2}$$

where $dist(a, b)$ is the distance function in Eq. 1.

Finally, a new cost function of autoencoder model is defined to make the representations of those rare words as close as possible to the representations of their corresponding words:

$$\min((C - \hat{C})^2 + (N - \hat{N})^2). \tag{3}$$

B. Nearest Neighbor Search Strategy

As shown in Fig. 2 and Fig. 3 , LSHWE uses locality sensitive hashing twice to generate the nearest neighbor list and the reconstructed nearest neighbor list, respectively.

Locality sensitive hashing is an approximate nearest neighbor search method, which is based on a simple idea that if two points are close with each other, then after a "projection" operation, these two points will remain close with each other. It has two steps. Firstly, since locality sensitive hashing is based on randomized space partitions, it uses a family $\mathcal{F}$ of hash functions [29] to perform dimension reduction and divide the inputs into "buckets" at the same time. Secondly, it searches the nearest neighbors in the "buckets" that have the target item.

In order to ensure that there is a high probability to have similar inputs be put into the same "bucket", the family $\mathcal{F}$ of hash functions needs to meet the following requirements: Given four parameters $r_1 > r_2$ and $p_1 > p_2$, a family $\mathcal{F}$ is said to be (r_1, r_2, p_1, p_2)-sensitive for a similarity measure $sim(x, y)$ if $\mathbf{Pr}_{h\in\mathcal{F}}[h(x) = h(y)] \geq p_1$ when $sim(x, y) \geq r_1$ and $\mathbf{Pr}_{h\in\mathcal{F}}[h(x) = h(y)] \leq p_2$ when $sim(x, y) \leq r_2$ [28].

978-1-7281-6926-2/20/$31.00 ©2020 IEEE

In this paper, we use the random hyperplane technique [30] to approximate the cosine distance between input vectors. Previous research found that random hyperplane technique can gives a family of hash functions $\mathcal{F}$ [29].

IV. Experiments

In this section, we empirically evaluated LSHWE from three aspects: effectiveness of LSHWE on cyberbullying detection task, algorithm efficiency and parameter sensitivity.

A. Experiment Setup

We use three datasets to validate LSHWE, as shown in Table I. Those datasets are randomly sampled from a Twitter dataset [31]. The parameter of rare words is set to be 2, i.e. $b = 2$.

TABLE I
Detailed Information of the Datasets

	Dataset1	Dataset2	Dataset3
Number of Tweets	4,000	6,000	8,000
Number of Bullying Tweets	800	1,200	1,600
Size of Corpus	7,928	10,087	11,876
Number of Rare Words	4,775	5,963	6,803
Rare Word Rate	60.23%	59.12%	57.28%

Before learning the representations of each word, we do a data preprocessing procedure to normalize its content, which includes the removal of: (a) all non-alphanumeric characters; (b) Web links; (c) stop words according to the default stop words corpus in the Natural Language Toolkit (NLTK). Then, we apply stemming action to reduce the word inflection.

The proposed word embedding method is implemented using TensorFlow. We use a 5-fold cross-validation for learning and testing. In each trial, we randomly select 80% of the data as the training dataset and use the rest as the testing dataset. Four metrics are used to measure the quality of the results, i.e. precision, recall, F1 score and running time.

All the experiments are executed on a personal computer with macOS Mojave, 2.5GHz Intel Core i7, and 16-GB memory.

B. Effectiveness of LSHWE on Cyberbullying Detection Task

We design an experiment to investigate the effectiveness of LSHWE in cyberbullying detection task. The methods we compared are three state-of-the-art word embedding methods: (a) GloVe [22], which uses a global log-bilinear regression model to unsupervised word representation learning and can extract the global statistic information; (b) Word2vec [21], which has two models: CBOW and Skip-gram, the authors' note [32] shows that CBOW is faster while Skip-Gram can achieve better representations for rare words; and (c) SSWE [33], which encodes sentiment information in the representation of words to solve the sentiment classification problem.

Fig. 4 compares the cyberbullying detection performance (i.e. F1 score) with different embedding methods on three datasets. The architecture of cyberbullying detector is shown in Fig. 5, we choose Long Short-Term Memory (LSTM) as the detector and the dropout rate is 0.2. From Fig. 4, we find that the detection performance of these three datasets shows an increasing trend. This may be because the detector can capture more latent features with the increase of the number of tweets, and those latent features can improve the detection accuracy.

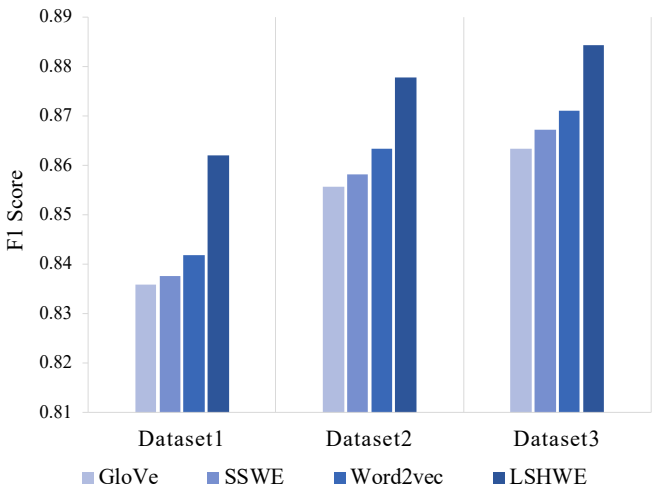

Fig. 4. Effect of Different Word Embedding Methods on Three Datasets.

Fig. 5. The Architecture of Cyberbullying Detector.

Especially, LSHWE performs the best on all datasets, followed by Word2vec, SSWE and GloVe. Besides, with the increase of the rare word rate, the detection performance gaps between LSHWE and the other methods become more and more obvious. This demonstrates that LSHWE can alleviate "deliberately obfuscated words" problem.

Since Fig. 4 only considers the case when the detector is LSTM, we design a new experiment to explore the detection performance of different word embedding methods when using different detectors. We choose seven detectors that are often used in cyberbullying detection task, including four machine learning-based cyberbullying detectors: Support Vector Machine (SVM), Naive Bayes (NB), Logistic Regression (LR), and Random Forest (RF); and three deep learning-based cyberbullying detectors: Long Short-Term Memory (LSTM), Bidirectional LSTM (BLSTM) and Bidirectional LSTM with attention (BLSTM_att). Note that the architectures of those three deep learning-based cyberbullying detectors are the same (as shown in Fig. 5), but it uses BLSMT or BLSTM_att instead of LSTM. Take dataset1 as an example, the results are shown in Fig. 6.

We find that LSHWE achieves the best detection result on most of these detectors. Only when using Support Vector Machine and Random Forest as the detector, the F1 score of Word2vec is slightly better than that of LSHWE. This further proves the effectiveness of LSHWE on cyberbullying detection

978-1-7281-6926-2/20/$31.00 ©2020 IEEE

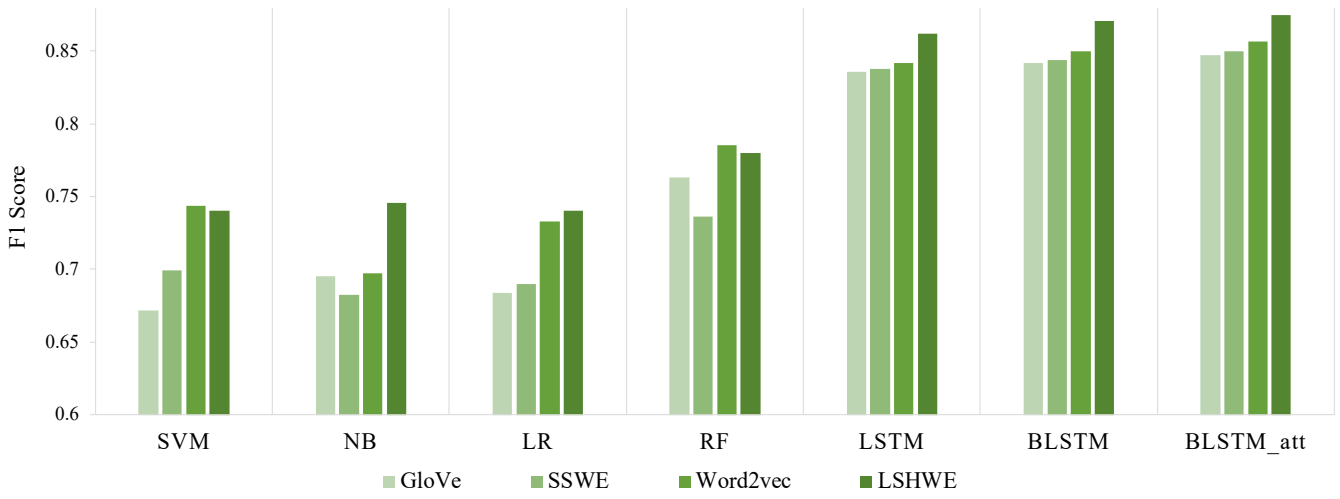

Fig. 6. Effect of Different Word Embedding Methods on Different Cyberbullying Detectors.

task.

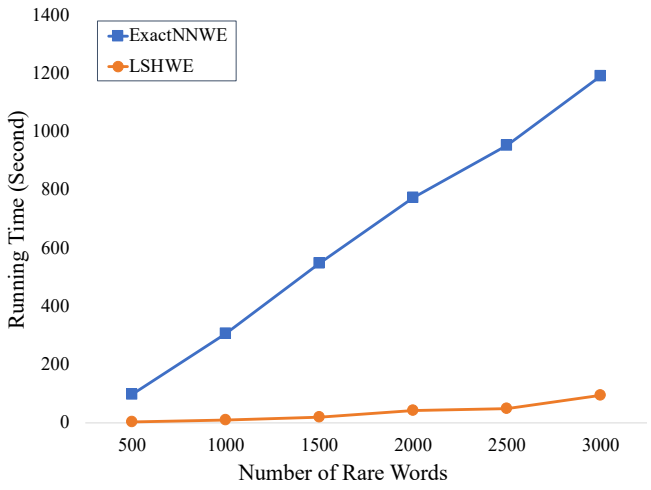

Fig. 7. Running Time of Two Similarity-Based Word Embedding Methods with Different Nearest Neighbor Search Strategy.

C. Efficiency of LSHWE

As we mentioned earlier, the reason for using locality sensitive hashing to search the nearest neighbors for each rare word is to improve the algorithm efficiency. So we design an experiment to verify the efficiency of LSHWE.

Exact NN-based word embedding method (ExactNNWE) is also a similarity-based word embedding method which has the same framework with LSHWE, except it uses the exact nearest neighbor search method to search the nearest neighbors instead of locality sensitive hashing.

Fig. 7 shows the running time (second) of these two similarity-based word embedding methods. Cosine Similarity is used as the distance function in these two word embedding methods. The dimension of the latent word vector and the number of nearest neighbors are set to be 25 and 4, respectively. We find the running time of ExactNNWE is far more than that of LSHWE when increasing the number of rare words, e.g. when the number of rare words reaches 3000, LSHWE takes around two minutes to learn word representations while ExactNNWE takes ten times longer than LSHWE. This shows that LSHWE is a highly efficient algorithm which can deal with large-scale datasets.

D. Parameter Sensitivity

Next, we investigate the detection performance with respect to four parameters: dimension of the latent word vector d, distance function, number of nearest neighbors k and hash size. Take dataset1 as an example, the cyberbullying detector is LSTM.

1) Dimension of the Latent Word Vector d: Fig. 8 reports the detection performance using latent word vectors with different dimension. The distance function is Cosine Similarity, the number of nearest neighbors and the hash size are set to be 4 and 5, respectively.

As shown in Fig. 8, F1 score increases with the increase of the latent word vector's dimension and tends to saturate once the dimension reaches around 20, where the F1 score has exceeded 0.85. This means the word vectors generated by LSHWE can achieve a high detection performance with a low dimension. Since the increase of the dimension of the input would result in more parameters in the LSTM detector, which requires more computation time, LSHWE can improve the detection efficiency to some extent.

2) Distance Function: We choose three distance functions which are often used to measure the similarity between two vectors: Euclidean Distance, Cosine Similarity and Manhattan Distance. Given two vectors A and B with dimension n, these distance functions can be described as follows:

978-1-7281-6926-2/20/$31.00 ©2020 IEEE

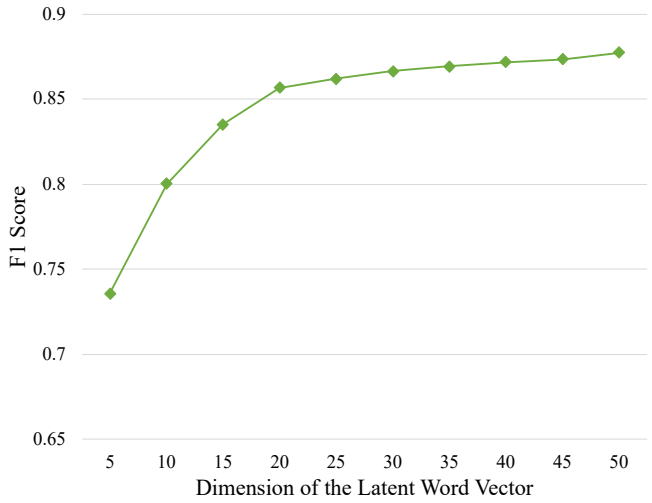

Fig. 8. Detection Performance Using Latent Word Vectors with Different Dimension.

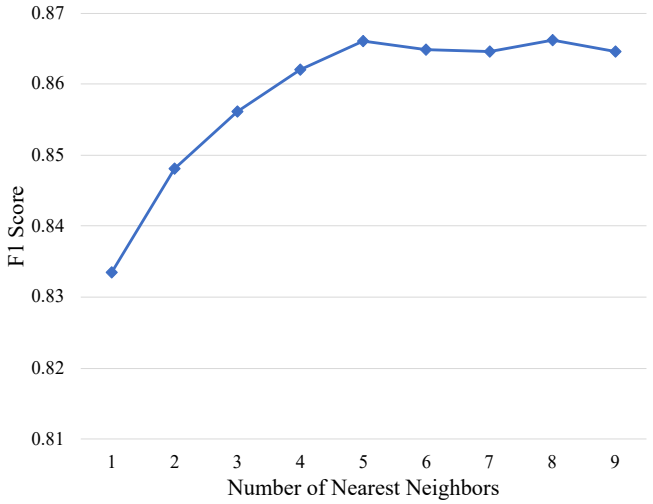

Fig. 9. Detection Performance Using Different Top-k Neighbors-Based Word Vectors.

$$Euclidean\ Distance = \sqrt{\sum_{i=1}^{n}(A_i - B_i)^2}, \quad (4)$$

$$Cosine\ Similarity = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}}, \quad (5)$$

$$Manhattan\ Distance = \sum_{i=1}^{n} |A_i - B_i|. \quad (6)$$

Table II reports the detection performance (i.e. precision, recall and F1 score) using different latent word vectors which are generated according to different distance functions. The dimension of the latent word vector, the number of nearest neighbors and the hash size are set to be 25, 4 and 5, respectively.

We observe that Cosine Similarity-based word embedding method achieves the best results, followed by Euclidean Distance-based word embedding method and Manhattan Distance-based word embedding method. However, since the computational complexity of Cosine Similarity is higher than the other two distance functions, the running time of Cosine Similarity-based word embedding method is also a little bit longer.

TABLE II
DETECTION PERFORMANCE USING DIFFERENT DISTANCE FUNCTION-BASED WORD VECTORS

	Precision	**Recall**	**F1 Score**
Cosine Similarity-Based Word Vectors	0.8605	0.8653	0.8629
Euclidean Distance-Based Word Vectors	0.8562	0.8551	0.8556
Manhattan Distance-Based Word Vectors	0.8506	0.8435	0.8470

3) Number of Nearest Neighbors: Fig. 9 shows the detection performance when using different word vectors which are generated based on different top-k nearest neighbors. The distance function is Cosine Similarity, the dimension of the latent word vector and the hash size are set to be 25 and 5, respectively.

We observe that increasing the number of nearest neighbors could improve the detection performance at the beginning, but F1 score wavers and shows a downward trend after the number of nearest neighbors reaches 5. This is because those "rare words" only have a little contextual information, so, the increase of the number of nearest neighbors would result in more noise, which consequently decreases the detection accuracy.

4) Hash Size: The first step in Locality Sensitive Hashing is using hash functions to map an item or items in multidimensional coordinate space to a scalar value. The hash size determines the number of planes, which is used to encode the data points. The more planes, the more buckets, the fewer items in each bucket.

Fig. 10 shows the detection performance when using different hash size-based word vectors in cyberbullying detection. The distance function is Cosine Similarity, the dimension of the latent word vector and the number of nearest neighbors are set to be 25 and 4, respectively.

From Fig. 10, we find that the increment of hash size makes detection performance increase first, but F1 score turns down when the hash size reaches 8. This is because that at the beginning, there are few buckets, and in each bucket, it has a large number of items, so it has some noise data when searching for the top-k nearest neighbors. With the increase of hash size, those noise data gradually decreases. When the hash size beyond a certain threshold, there will have too many buckets and some buckets have less than k items in it, so we cannot find the top-k nearest neighbors for some rare words.

978-1-7281-6926-2/20/$31.00 ©2020 IEEE

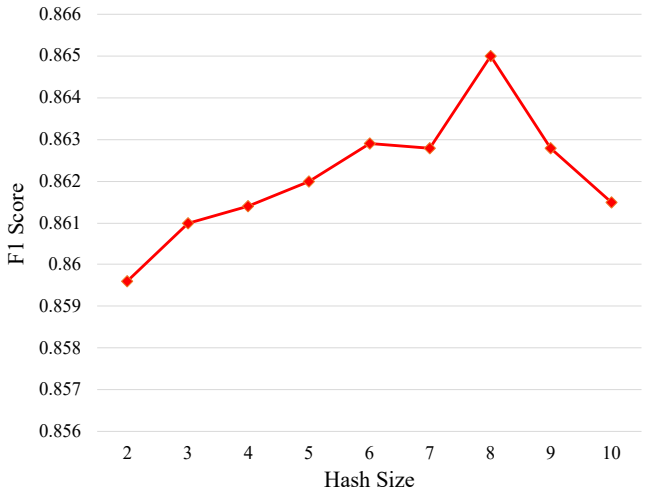

Fig. 10. Detection Performance Using Different Hash Size-Based Word Vectors.

Moreover, the fewer items in the buckets, the less time will be need to spend on searching for the top-k nearest neighbors.

V. Conclusion and Future Work

In this paper, we propose a similarity-based word embedding method LSHWE to solve the "deliberately obfuscated words" problem in cyberbullying detection task. LSHWE has two steps. Firstly, for a given corpus, it generates: (a) a co-occurrence matrix C; (b) a rare word list R; (c) a nearest neighbor list NL obtained by locality sensitive hashing; and (d) a nearest neighbor matrix N. Secondly, an LSH-based autoencoder is used to learn the word vectors according to C and N. The proposed embedding method has two characteristics: (1) LSHWE can represent well on rare words. LSHWE is a global similarity-based word embedding method thus the representations of rare words learnt through LSHWE can be as close as possible to their corresponding words' representations; and (2) LSHWE is a highly efficient algorithm. This method uses an approximate nearest neighbor search method to search the top-k nearest neighbors instead of exact nearest neighbor search methods, which can greatly reduce the running time.

We design experiments from three aspects: effectiveness of LSHWE on cyberbullying detection task, algorithm efficiency and parameter sensitivity. Experiment results demonstrate that LSHWE can alleviate the "deliberately obfuscated words" problem and is highly efficient on large-scale datasets.

Our future work will focus on solving the "deliberately obfuscated words" problem from other perspectives, such as morphology. Besides, all rare words are considered as the input of LSHWE, but not all of them are deliberately obfuscated words. So we will use a method to extract deliberately obfuscated words among these rare words before learning word vectors in our future work.

Acknowledgment

This work is supported by the Graduate Scientific Research and Innovation Foundation of Chongqing (No. CYS19028), the Technological Innovation and Application Program of Chongqing (Project No. cstc2018jszx-cyzdX0081) and a Small Project Grant of China Studies Centre, The University of Sydney, Australia.

References

[1] Stopbullying.gov. (2019) What is cyberbullying? [Online]. Available: https://www.stopbullying.gov/cyberbullying/what-is-it/index.html

[2] Ditchthelabel. (2014) The annual cyberbullying survey. [Online]. Available: http://www.ditchthelabel.org/downloads/the-annual-cyberbullying-survey-2013.pdf

[3] S. Tulkens, L. Hilte, E. Lodewyckx, B. Verhoeven, and W. Daelemans, "A dictionary-based approach to racism detection in dutch social media," *arXiv preprint arXiv:1608.08738*, 2016.

[4] J. Bayzick, A. Kontostathis, and L. Edwards, "Detecting the presence of cyberbullying using computer software," in *Proceedings of the 3rd annual ACM web science conference*. Evanston, United States: ACM, 2011.

[5] R. Zhao, A. Zhou, and K. Mao, "Automatic detection of cyberbullying on social networks based on bullying features," in *Proceedings of the 17th International Conference on Distributed Computing and Networking*. Singapore, Singapore: ACM, 2016.

[6] K. Radoslaw, P. Michal, R. Rafal, and A. Kenji, "Recognizing and converting cockney rhyming slang for cyberbullying and crime detection," in *Proceedings of Language Sense on Computers IJCAI 2016 Workshop*. New York, United States: IJCAI, 2016.

[7] G. K. Pitsilis, H. Ramampiaro, and H. Langseth, "Detecting offensive language in tweets using deep learning," *arXiv preprint arXiv:1801.04433*, 2018.

[8] V. K. Singh, Q. Huang, and P. K. Atrey, "Cyberbullying detection using probabilistic socio-textual information fusion," in *Proceedings of the 2016 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*. San Francisco, United States: IEEE, 2016.

[9] M. Dadvar and A. Niamir, "Adopting maxent to identification of bullying incidents in social networks," in *Proceedings of the 27th International Workshop on Database and Expert Systems Applications*. Porto, Portugal: IEEE, 2016.

[10] L. Cheng, J. Li, Y. N. Silva, D. L. Hall, and H. Liu, "Xbully: Cyberbullying detection within a multi-modal context," in *Proceedings of the 12th ACM International Conference on Web Search and Data Mining*. Melbourne, Australia: ACM, 2019.

[11] B. Gambäck and U. K. Sikdar, "Using convolutional neural networks to classify hate-speech," in *Proceedings of the 1st workshop on abusive language online*. Vancouver, Canada: ACL Anthology, 2017.

[12] A.-M. Founta, D. Chatzakou, N. Kourtellis, J. Blackburn, A. Vakali, and I. Leontiadis, "A unified deep learning architecture for abuse detection," *arXiv preprint arXiv:1802.00385*, 2018.

[13] P. Badjatiya, S. Gupta, M. Gupta, and V. Varma, "Deep learning for hate speech detection in tweets," in *Proceedings of the 26th International Conference on World Wide Web Companion*. Perth, Australia: International World Wide Web Conferences Steering Committee, 2017.

[14] A. Bakarov, "A survey of word embeddings evaluation methods," *arXiv preprint arXiv:1801.09536*, 2018.

[15] C. Nobata, J. Tetreault, A. Thomas, Y. Mehdad, and Y. Chang, "Abusive language detection in online user content," in *Proceedings of the 25th international conference on world wide web*. Montreal, Canada: International World Wide Web Conferences Steering Committee, 2016.

[16] A. Schmidt and M. Wiegand, "A survey on hate speech detection using natural language processing," in *Proceedings of the 5th International Workshop on Natural Language Processing for Social Media*. Valencia, Spain: ACL Anthology, 2017.

[17] C. Lin, D. Liu, W. Pang, and Z. Wang, "Sherlock: A semi-automatic framework for quiz generation using a hybrid semantic similarity measure," *Cognitive computation*, vol. 7, no. 6, pp. 667–679, 2015.

[18] D. Liu and C. Lin, "Sherlock: a semi-automatic quiz generation system using linked data." in *International Semantic Web Conference (Posters & Demos)*. Riva del Garda, Italy: Citeseer, 2014.

[19] P. Wang, Y. Qian, F. K. Soong, L. He, and H. Zhao, "Word embedding for recurrent neural network based tts synthesis," in *Proceedings of the 2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. South Brisbane, Australia: IEEE, 2015.
[20] S. Wang, W. Zhou, and C. Jiang, "A survey of word embeddings based on deep learning," *Computing*, vol. 102, no. 3, pp. 717–740, 2020.
[21] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *arXiv preprint arXiv:1301.3781*, 2013.
[22] J. Pennington, R. Socher, and C. D. Manning, "Glove: Global vectors for word representation," in *Proceedings of the 19th Conference on Empirical Methods in Natural Language Processing*. Doha, Qatar: ACL Anthology, 2014.
[23] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Minneapolis, Minnesota: ACL Anthology, 2019.
[24] R. Li, C. Lin, M. Collinson, X. Li, and G. Chen, "A dual-attention hierarchical recurrent neural network for dialogue act classification," in *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*. Hong Kong: ACL Anthology, 2019.
[25] R. Mao, C. Lin, and F. Guerin, "Word embedding and wordnet based metaphor identification and interpretation," in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Melbourne, Australia: ACL Anthology, 2018.
[26] M. R. Abbasifard, B. Ghahremani, and H. Naderi, "A survey on nearest neighbor search methods," *International Journal of Computer Applications*, vol. 95, no. 25, pp. 39–52, 2014.
[27] A. Andoni, P. Indyk, and I. Razenshteyn, "Approximate nearest neighbor search in high dimensions," *arXiv preprint arXiv:1806.09823*, 2018.
[28] P. Indyk and R. Motwani, "Approximate nearest neighbors: towards removing the curse of dimensionality," in *Proceedings of the 30th annual ACM symposium on Theory of computing*. Dallas, United States: ACM, 1998.
[29] M. S. Charikar, "Similarity estimation techniques from rounding algorithms," in *Proceedings of the 34th annual ACM symposium on Theory of computing*. Montreal, Canada: ACM, 2002.
[30] A. Andoni and P. Indyk, "Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions," in *Proceedings of the 47th annual IEEE symposium on foundations of computer science*. Berkeley, United States: IEEE, 2006.
[31] Z. Waseem and D. Hovy, "Hateful symbols or hateful people? predictive features for hate speech detection on twitter," in *Proceedings of the NAACL student research workshop*. San Diego, United States: ACL Anthology, 2016.
[32] code.google.com. (2013) word2vec. [Online]. Available: https://code.google.com/archive/p/word2vec/
[33] D. Tang, F. Wei, N. Yang, M. Zhou, T. Liu, and B. Qin, "Learning sentiment-specific word embedding for twitter sentiment classification," in *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Baltimore, United States: ACL Anthology, 2014.

978-1-7281-6926-2/20/$31.00 ©2020 IEEE