A Multiview Clustering Framework for Detecting Deceptive Reviews

Yubao Zhang [a], Haining Wang [b] and Angelos Stavrou [b]
[a] *Department of Electrical and Computer Engineering, University of Delaware, DE, U.S.*
[b] *Department of Electrical and Computer Engineering, Virginia Tech, VA, U.S.*

Abstract. Online reviews, which play a key role in the ecosystem of nowadays business, have been the primary source of consumer opinions. Due to their importance, professional review writing services are employed for paid reviews and even being exploited to conduct opinion spam. Posting deceptive reviews could mislead customers, yield significant benefits or losses to service vendors, and erode confidence in the entire online purchasing ecosystem. In this paper, we ferret out deceptive reviews originated from professional review writing services. We do so even when reviewers leverage a number of pseudonymous identities to avoid the detection. To unveil the pseudonymous identities associated with deceptive reviewers, we leverage the multiview clustering method. This enables us to characterize the writing style of reviewers (deceptive vs normal) and cluster the reviewers based on their writing style. Furthermore, we explore different neural network models to model the writing style of deceptive reviews. We select the best performing neural network to generate the representation of reviews. We validate the effectiveness of the multiview clustering framework using real-world Amazon review data under different experimental scenarios. Our results show that our approach outperforms previous research. We further demonstrate its superiority through a large-scale case study based on publicly available Amazon datasets.

Keywords: Multiview Clustering, Deceptive Reviews

Online reviews of commercial products and services extensively impact consumers' decision making that goes beyond purchasing a specific product extending to trustworthiness of its brand. Most customers read online reviews and trust them explicitly because they treat them as personal recommendations from fellow shoppers. Thus, in many cases, online reviews are a crucial factor in a customer's decision before committing to an online purchase. Prior studies show that 80% of consumers reverse product purchase decisions after reviewing negative online reviews, and 87% affirm a purchase decision based on positive online reviews [1].

The importance of online reviewing for product purchasing has given rise to professional reviewing writing services both to praise a product but also to damage the image of a competitive one. In literature, deceptive opinion spam is defined as fictitious opinions that have been deliberately written to mislead consumers but sound authentic. This is especially true for deceptive reviews crafted by professional review writing services because they are usually very eloquent and designed to be convincing. Due to their nature, deceptive reviews have more impact and tend to be more misleading. Moreover, deceptive review writers have experience in faking reviews to obtain free products and compensation. To cover their true identity and avoid detection by the e-commerce sites, professional reviewers rely on a number of pseudonymous identities. The need for voluminous deceptive online reviews gave rise to online brokers - such as review groups/clubs. These online groups solicit and coordinate deceptive review writers with parties interesting in hiring them to boost the popularity of their products in online stores or damage the competition.

Exposing deceptive reviews at scale is a challenging problem primarily because pseudonymous identities have become hard to uncover by merely looking at their profiles [2–4]. However, we posit that for

those deceptive reviews by the same writer and with a common objective, they inevitably share some similarities with regards to the writing style. By revealing the common writing style of deceptive reviews, we show that we can shed light on the detection of deceptive reviews.

In this paper, we attempt to detect deceptive reviews by leveraging their shared similarities in the writing style. On one hand, pseudonymous identities that are controlled by the same deceptive review writer apparently possess a similar writing style. On the other hand, the deceptive reviews bear the same objective for opinion deception of a certain product or a collection of products, which could also lead to some shared similarities in the writing style. For example, the deceptive reviews could be exaggerated on the quality of a product.

As pseudonymous identities are under the control of a same deceptive review writer, there exist similarities in the writing style to some extent. To reveal the pseudonymous identities, we employ a multiview clustering method to cluster reviewers based on multiview data. Each review of a reviewer is treated as a *view* of the reviewer, since each review is related to a different product. We validate the efficacy of the multiview clustering framework through extensive experiments, showing that our multiview clustering method outperforms K-means and hierarchical methods.

To identify the writing style of deceptive reviews, we explore different neural network models and choose the neural network model with the best performance based on two public datasets [5, 6]. Our experiments indicated that the convolutional neural network approach outperforms other neural network models, and can achieve approximately 90% accuracy on both datasets. Therefore, we selected the convolutional neural networks to generate a representation that best captures the deceptive writing style. We integrate the representation into the multiview clustering framework as a deceptive view. Our evaluation shows that the deceptive view can improve the performance of clustering on multiple other datasets.

Armed with the modified multiview clustering framework, we apply our deceptive view approach on large scale Amazon datasets [7, 8] as a case study. We detected suspicious clusters associated with deceptive review writers. In addition, we examined the reviewer clusters using different criteria including reviewers' lifespan, feedback, and overall rating. Our evaluation results demonstrate that the proposed detection approach is effective to filter out the suspicious clusters for further examination.

The remainder of this paper is structured as follows. Section 1 briefs the background. Section 2 presents our proposed framework. Section 3 evaluates the proposed framework. Section 4 demonstrates a case study on large scale Amazon datasets. Section 6 surveys the related work, and finally, Section 7 concludes the paper.

1. Understanding Deceptive Reviews

Since online reviews play a key role in e-commerce, merchants always desire positive reviews to boost their business. Meanwhile, deceptive review writers are willing to write positive reviews for compensation or free products. There are brokers (*e.g.*, review group) in the middle to connect the merchants and deceptive review writers. The merchants could also launch campaigns in online social networks [9]. As the brokers play an important role in deceptive review distribution, understanding the brokers will be very helpful for the detection of deceptive review writers.

Take review group as an example, through the process of review group, merchants can obtain reviews that are tagged as verified purchase by Amazon and thus seem more authentic. These reviews are called incentivized reviews, which are prohibited in Amazon. Fig. 1 shows how an Amazon review group works. (1) The merchant posts a review request in a broker (*i.e.*, review group), and then (2) the broker

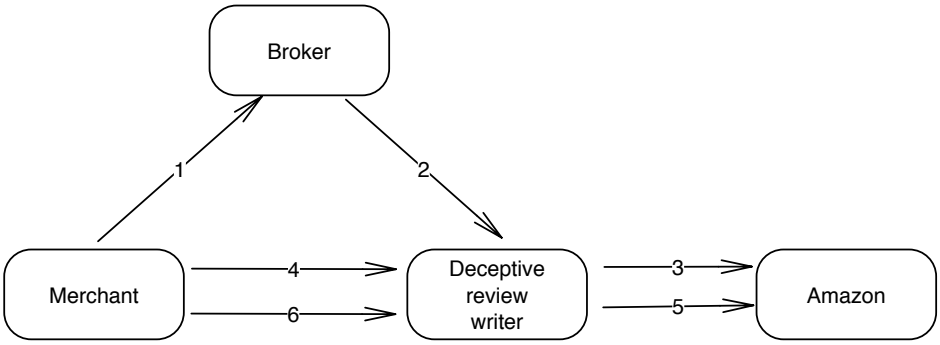


Fig. 1. The process of review group.

notifies the deceptive review writer. (3) If the deceptive review writer accepts the task, he/she purchases the product from Amazon like a normal customer does. After that, (4) the merchant assigns the crafted review to the deceptive review writer or ask the deceptive review writer to craft a review, and then (5) the deceptive review writer posts it to Amazon. Finally, (6) the merchant makes a payment to the deceptive review writer.

To understand how merchants, deceptive review writers, and brokers work together, we joined several Amazon review groups and found that most of deceptive review writers have more than one accounts. This is because Amazon has strict review policy, and it suspends the accounts of both merchants and customers that are involved in the deceptive reviews. Deceptive review writers operate multiple accounts in order to evade the detection and suspension. In this work, we focus on the detection of deceptive reviews that are published by deceptive review writers. We cluster the review accounts based on the writing style in a large scale, and then we check the suspicious clusters that are likely associated with deceptive review writers. Note that each suspicious cluster of review accounts are connected to a same deceptive review writer. For ease of presentation, we refer to a review account as a reviewer in the rest of this paper.

2. Detection Framework

In this section, we attempt to detect deceptive reviews by identifying suspicious deceptive reviewers based on their shared similarities in the writing style. We cluster reviewers to detect the suspicious clusters associated with deceptive reviewers based on the writing style. Here we explore the neural network models to capture the deceptive writing style. For example, deceptive reviews need to exaggerate a specific product and possess similar exaggerative words.

Challenges: The challenges in finding the pseudonymous identities associated with deceptive review writers are listed as follows:

- We have no prior knowledge of the writing style of deceptive reviewers, and we need to cluster the reviewers in a large scale and further check the suspicious clusters. Zero knowledge of deceptive reviewers precludes the use of some existing methods, such as authorship attribution, which requires the knowledge to build the model of each author.
- The reviews of each account are related to different products and are independent of each other, and thus the clustering of reviewers must consider each single review.

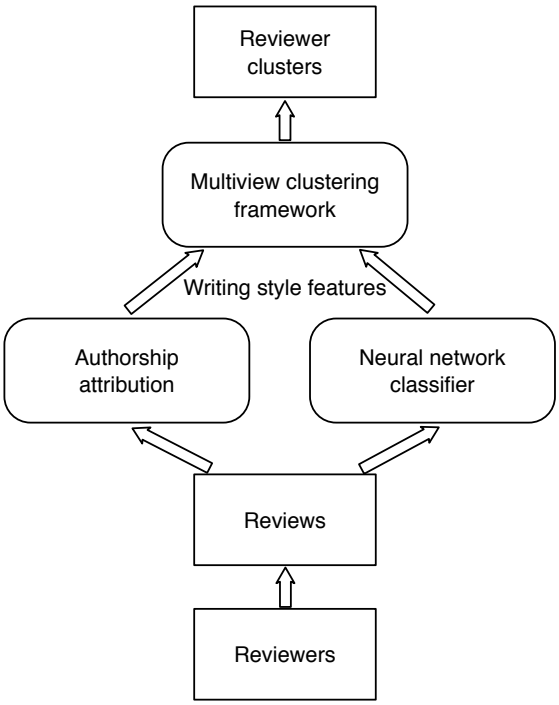


Fig. 2. Multiview clustering method.

- The deceptive writing style (*e.g.*, exaggeration) is difficult to be quantitatively captured and employed in the detection.

To address the challenges, we employ a multiview clustering method (as shown in Fig. 2) to cluster the reviewers, as we can take each review into consideration. The multiview clustering method first computes the similarity of two reviewers by comparing each of their reviews. Here, we regard a review as a *view* of the reviewer. A local similarity network of reviewers are created by computing the similarity with one view of each reviewer. Then, local similarity networks are fused to generate the overall similarity network using the multiview clustering method. The overall similarity score is utilized to cluster the reviewers. To capture the deceptive writing style of deceptive reviews, we extract the features from neural network models and regard the features as another *view* of the reviewer. We will integrate the deceptive view into the multiview clustering framework later on.

2.1. Multiview Clustering

Since reviews of a reviewer are related to different products, each review needs to be considered separately when comparing reviewers. To this end, we employ the multiview clustering method [10] to perform the clustering of reviewers, in order to detect the clusters of accounts that are associated with deceptive review writers. The multiview clustering can combine multiview features with data information and search for consistent clusterings across the different views.

View: We define each review of a reviewer as a *view*, since each review is associated with a particular product and thus provides additional information for profiling the writing style of the reviewer. Moreover, we consider additional views from neural network models.

Similarity network: We extract one review from each reviewer and compute the similarity matrix across all the reviewers. The similarity matrix is called the local similarity network, which represents the similarity score between every two reviewers. Since the similarity score is computed based on only one review from each reviewer, the similarity network represents the similarity measure of one view. After computing across different views, we obtain a collection of similarity networks. The multiview clustering method is utilized to fuse the similarity networks to search for the consistent clusterings across different views. Figure 3 illustrates an example of the multiview clustering with nine reviewers, each of whom has two reviews.

Fusing Similarity Networks: Suppose we have N reviewers and each reviewer has M reviews. A similarity network can be represented as $G = (V, E)$, where the vertices V correspond to the reviewers $\{a_1, \cdots, a_N\}$ and the edges E indicate how similar the reviewers are. The edge weights are represented by an similarity matrix $W_{N\times N}$ with $W_{i,j}$ indicating the similarity between reviewers a_i and a_j, which is computed as follows:

$$W_{i,j} = \exp(-\frac{d^2(a_i, a_j)}{\mu\varepsilon_{i,j}}), \tag{1}$$

where $d(a_i, a_j)$ denotes the Euclidean distance between a_i and a_j, μ is a hyperparameter that can be empirically set, and $\varepsilon_{i,j}$ is used to handle the scaling problem. Here $\varepsilon_{i,j}$ is defined as

$$\varepsilon_{i,j} = \frac{mean(d(a_i, a_{N_i})) + mean(d(a_j, a_{N_j})) + d(a_i, a_j)}{3},$$

where $mean(d(a_i, a_{N_i}))$ is the average distance between a_i and its neighbors.

To fuse the similarity matrices from different similarity networks, a full and sparse kernel is defined on the vertex set V. The full kernel is a normalized weight matrix P with P_{ij} defined as follows:

$$P(i, j) = \begin{cases} \frac{W_{i,j}}{2\sum_{k\neq i} W_{i,k}}, & j \neq i \\ 1/2, & j = i. \end{cases} \tag{2}$$

This normalization makes $\sum_j P_{i,j} = 1$ hold. The sparse kernel only considers local similarity and sets remote ones to zero. Given a graph G, local similarity is measured with K-nearest neighbors. Let KNN_i denote K-nearest neighbors of a_i, including a_i itself. The sparse kernel is a local affinity defined as:

$$S(i, j) = \begin{cases} \frac{W_{i,j}}{\sum_{k\in KNN_i} W_{i,k}}, & j \in KNN_i \\ 0, & \text{otherwise.} \end{cases} \tag{3}$$

Therefore, P contains the full similarity information, while S only has the local similarity information of K-nearest neighbors.

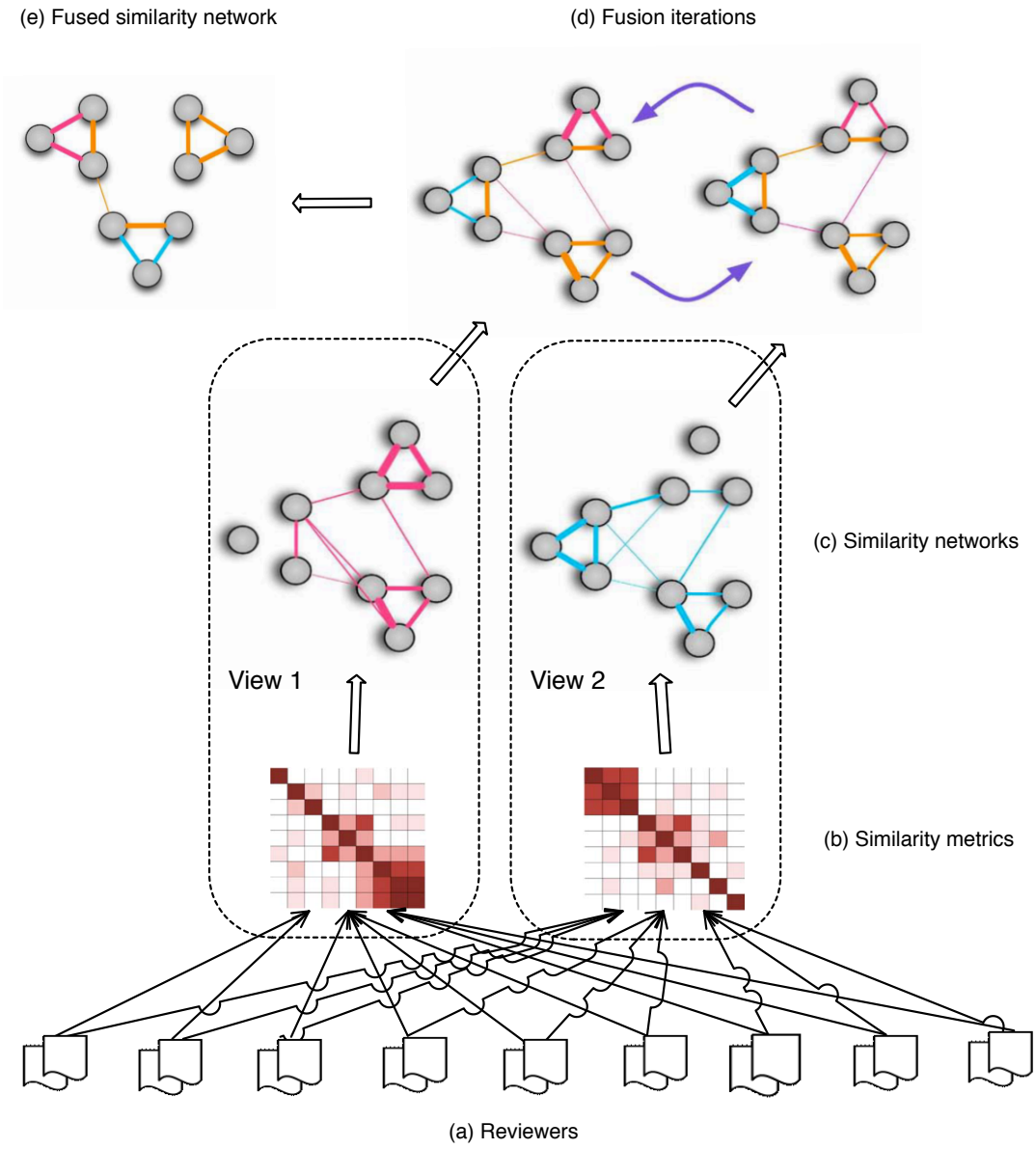


Fig. 3. Fusing similarity networks

Without loss of simplicity and generality, we consider the setting of fusing two similarity networks through a full sparse kernel as shown in Figure 3. Let $W^{(1)}$ and $W^{(2)}$ denote the similarity matrices of two similarity networks. We can calculate $(P^{(1)}, P^{(2)})$ as in Equation 2 and $(S^{(1)}, S^{(2)})$ as in Equation 3. Let $P_0^{(1)}$ and $P_0^{(2)}$ represent the initial similarity matrices at $t = 0$. Similarity matrices $P^{(1)}$ and $P^{(2)}$ are updated iteratively as follows:

$$P_{t+1}^{(1)} = S^{(1)} \times P_t^{(2)} \times (S^{(1)})^T \tag{4}$$

$$P_{t+1}^{(2)} = S^{(2)} \times P_t^{(1)} \times (S^{(2)})^T. \tag{5}$$

This fusion process exchanges the similarity matrices in each iteration and generates two parallel interchanging diffusion processes. After n steps, the overall similarity matrix is computed as $P^{(c)} = \frac{1}{2}(P_n^{(1)} + P_n^{(2)})$. Note that we can easily extend the process to the case with M views, *i.e.*, each review has a view, by updating vth similarity matrix as follows:

$$P_{t+1}^{(v)} = S^{(v)} \times (\frac{\sum_{k \neq v} P_t^{(k)}}{m-1}) \times (S^{(v)})^T, v = 1, 2, \cdots, m. \tag{6}$$

Through the above process of fusion, we obtain the overall similarity matrix $P^{(c)}$, which can further be used to perform reviewer clustering.

2.2. Deceptive View

In [11], Ren *et al.* explored a neural network model to detect deceptive reviews by learning the document-level representation. Inspired by this work, we explore neural network models and produce a representation of each review to capture the deceptive writing style. The representation of a review is utilized to classify the review as being deceptive or not. Here, we take the representation produced by the neural network model as a deceptive view for multiview clustering. The neural network model has been proved to be capable of capturing some writing styles that are unique for deceptive reviews. Therefore, the deceptive view constructed via the neural network model represents the degree to which a review is related to deception.

Neural network models: We explore convolutional neural networks and recurrent neural networks to produce the deceptive view.

Convolutional Neural Networks: Reviews are tokenized using a plain word tokenizer. Words or characters are represented by non-sparse vectors, also known as *embeddings*. A sequence of words or characters is represented as a matrix, where each column corresponds to the embedding vector value. The convolutional layer applies a set of convolutional filters of different sizes. We use 128 convolution filters, each of which has length 3, 4 and 5 words or characters, and a batch size of 128 sentences. Then a max-over-time pooling layer performs maxing-over-time pooling operations over the output feature maps, where only the maximum value of each feature map is used. The max pooling outputs of each feature map are concatenated in a vector, which is used in the fully connected layer with dropout and softmax output to obtain the prediction result. The dropout rate is 0.5.

Recurrent Neural Networks: Since simple RNN suffers from learning long-term dependencies, we implement Long Short Term Memory (LSTM) networks. LSTM, a special kind of RNN, can memorize the information of long periods of time. The size of a hidden unit is 100 and the dropout rate is 0.5.

We also implement the *attention* mechanism on RNN to explicitly handle the hidden states of RNN. The attention mechanism enables the access to the hidden states of RNN. The mechanism that we implement retrieves a weighted combination of all hidden states, which can retain long-range dependency information of reviews.

3. Evaluation

In this section, we evaluate the multiview clustering framework with real Amazon review data [7]. We randomly choose a certain number (*e.g.*, 10, 100, and 1,000) of reviewers from the dataset. Each reviewer's reviews are divided into 10 equal segments and tagged to 10 different reviewers, such that we can have the ground truth that the reviews of these 10 reviewers are from the same reviewer and is able to evaluate the clustering results. For example, we select 1,000 reviewers and divide their reviews to construct a dataset with 10,000 reviewers. Here we have the true labels of reviewers serving as the ground truth for evaluation. Parameter K represents the number of nearest neighbors and is set to 10 by default, while μ is a hyperparameter defined in Equation 1 and is set to 0.5 by default.

Writing Style Features of Reviews: Table 1 summarizes the feature set, which is similar to that of features used in authorship attribution. We preprocess the reviews to remove URL and convert the text into lowercase. Note that stop words are not removed since they play an important role in capturing authors' writing style. The punctuations and other special characters that we count are *.?!,;:()"-'* and *#$%&*+/<=>@[]^_‘{|}~*, respectively, 32 in total. We use the same function words as listed in [12] (e.g., "the", "over", and "and").

Evaluation Metrics: First, we use Normalized Mutual Information (NMI) to measure the accuracy of clustering. NMI is a normalization of the Mutual Information (MI) score to scale the results between 0 and 1. If the score is 0, it means that there is no mutual information between true labels and predicted labels; while 1 means the perfect correlation between true labels and predicted labels.

Second, we use the silhouette score to measure the homogeneity of clusters. The silhouette score of reviewer i is calculated as $s_i = (b_i - a_i)/max(a_i, b_i)$, where a_i denotes the average dissimilarity for reviewer i to all other reviewers within the same cluster, and b_i means the distance between reviewer i and the nearest cluster that reviewer i is not a part of. The mean value of silhouette scores over all reviewers is used to measure how tightly grouped the reviewers are. If the silhouette score is close to 1, it means the reviewers are appropriately clustered.

Varying Cluster Number: To find the optimal cluster number, we use different cluster numbers for clustering in a set of 100 candidate reviewers, whose reviews are generated from 10 original reviewers. As Figure 4 shows, we find that NMI clearly reveals the optimal cluster number. Meanwhile, the change rate of the silhouette score is also helpful to indicate the optimal cluster number, since the silhouette score approaches stable when the cluster number is larger than the optimal one. Therefore, we assign the optimal cluster number to the following clustering experiments by default.

Comparing with Other Clustering Methods: We compare the multiview clustering method with other methods, including hierarchical and K-means clustering [13]. We use the agglomerative method and average linkage to build a hierarchy of clusters in the hierarchical clustering. K-means++ optimizes the seeds of clusters, while Principal Component Analysis (PCA) is used to perform dimensionality reduction before applying K-means.

The comparison is performed with a set of 100 candidate reviewers belonging to 10 clusters. For the multiview clustering method, we extract features at the review level and consider 10 views from 10

Table 1
Adopted features.

Category	Description	Count
Lexical		
	Digit frequency	10
	Special characters frequency	21
	Alphabet frequency	52
	Word n-gram features	200
Syntactic		
	Punctuation frequency	11
	Function words frequency	293
	POS tag n-gram features	200
Writing		
Density	Average number of characters per word	1
	Average number of syllables per word	1
	Average words per sentence	1
Readability		
	Flesch-Kincaid grade level	1
	Gunning fog index	1
	Yule's K measure	1
	Automated readability index	1
	Coleman-Liau index	1
	SMOG(Simple Measure of Gobbledygook) index	1
	LIX	1
	RIX	1

Table 2
Comparison of different clustering methods

Method	**NMI**	**Silhouette score**
Multiview	0.97	0.85
Hierarchical	0.57	0.24
K-means	0.87	0.19
K-means++	0.69	0.20
K-means(PCA)	0.85	0.21

reviews. However, for the hierarchical and K-means methods, we unite all reviews of each reviewer to become a document and extract features to perform clustering. Table 2 shows our experiment result. It demonstrates that the multiview clustering method outperforms the other two methods in both NMI and silhouette score, implying that it has higher accuracy and a model with better defined clusters.

Varying Reviewer Scale: To verify the performance of the multiview clustering method in different scales, we vary the reviewer scale from 100 to $10,000$. Figure 5 shows the results. We observe that both NMI and the silhouette score drop when the scale increases. However, when the scale reaches $10,000$,

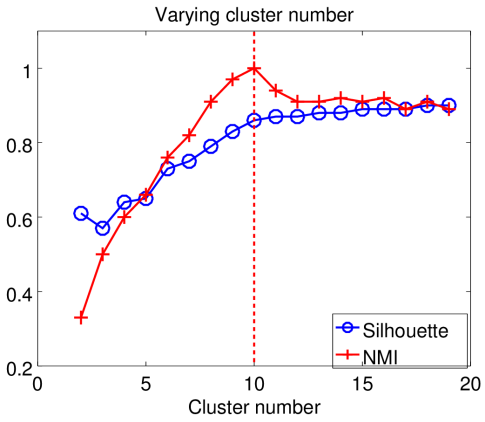


Fig. 4. We obtain an optimal result for K-10.

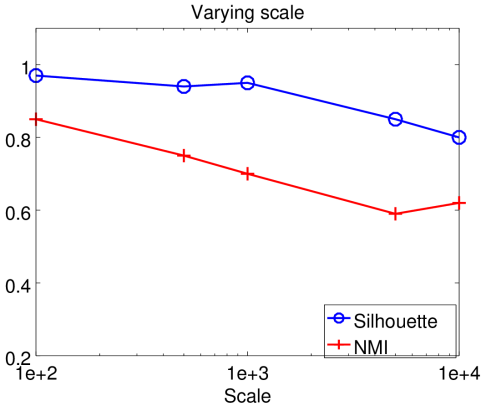


Fig. 5. Varying reviewer scale.

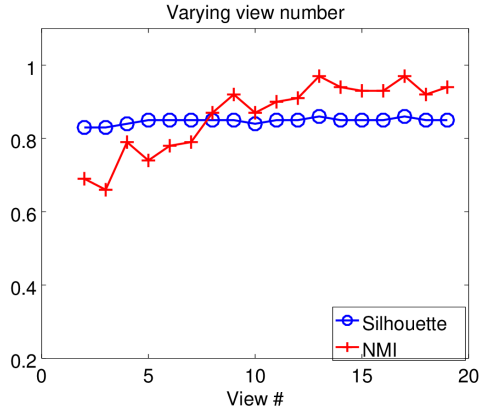


Fig. 6. Varying number of views.

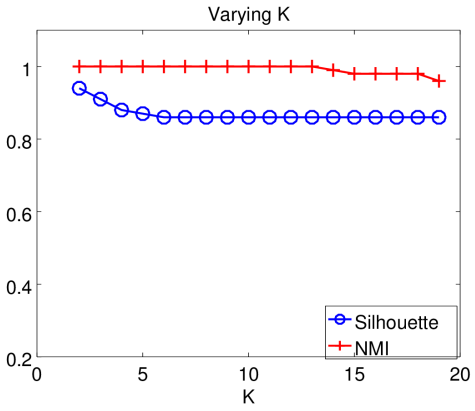


Fig. 7. Varying K, notice that the NMI is pretty stable for small numbers of K.

the performance is still tolerable with NMI of 0.62 and the silhouette score of 0.8. It implies that the multiview method scales well.

Varying Number of Views: As mentioned above, each view represents a review of each reviewer. We extract text features from these reviews and compute the similarity matrix of the view. Thus, varying view number means that a different number of reviews from each reviewer are used for clustering. In other words, it indicates how many reviews are needed to achieve acceptable performance. Figure 6 demonstrates the results when the view number is varied from 2 to 19, given a set of 100 candidate reviewers. We observe that NMI increases sharply after 5 and approaches stable when the view number reaches 13; while the silhouette score changes slightly. It implies that the multiview clustering method can achieve excellent performance with a considerably small number of reviews from each reviewer.

Varying K: We further investigate the impact of parameters upon the performance, including K and μ. We vary K and μ, respectively, given a set of 100 candidate reviewers. Figure 7 shows the results when K ranges from 2 to 19. NMI remains high and the silhouette score drops slightly as K increases. Meanwhile, NMI increases sharply and the silhouette score slightly decreases when varying μ, as shown in Figure 8.

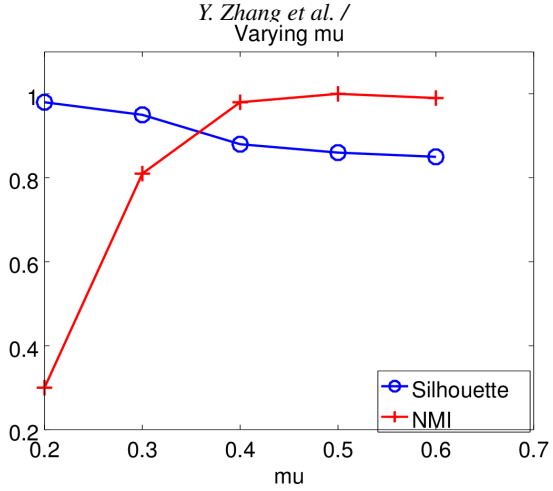


Fig. 8. Varying μ.

3.1. Exploring Neural Network Models

We explore different neural network models to determine which neural network model we utilize to generate the representation for deceptive view. Therefore, we search for the neural network model with the best performance in deception classification.

Datasets and Metrics: We use two datasets to evaluate different neural network models. One is introduced in [5, 14], called *TripAdvisor* dataset. The dataset contains 800 truthful reviews obtained from multiple websites and 800 deceptive reviews gathered using Amazon Mechanical Turk. Both truthful reviews and deceptive reviews contain equal positive reviews and negative reviews. The other dataset is collected from Yelp and first used in [6], called *Yelp* dataset. It includes 55, 794 reviews for a set of hotels in the Chicago area. The reviews filtered by the filtering algorithm of Yelp are considered as deceptive reviews, a total of 5, 078. Table 3 summarizes the datasets. In the following experiments, we randomly select an equal number of truthful reviews to balance the dataset.

Table 3
Statistics of datasets

Class	Dataset	# of reviews	# of unique words
Deceptive	TripAdvisor	800	4,180
	Yelp	5,078	19,264
Truthful	TripAdvisor	800	4,934
	Yelp	58,716	273,229

We adopt ten-fold cross-validation. In each fold, both datasets are split randomly into training/tuning/testing sets with a ratio of 80/10/10. The tuning set is mainly utilized for optimizing the hyper-parameters. We employ accuracy, precision, and recall to measure the performance of the neural network models.

Word Embeddings vs. Character Embeddings: We take into consideration both word embeddings and character embeddings for generating the deceptive view. We take GloVe 50 dimensional embeddings as word pretrained embeddings, which are trained on 6 billion words from Wikipedia and web text [15]. Word embeddings are fine-tuned during the training process. We use the average of all the

Table 4
Comparison of different deep learning methods

Models	Dataset	Accuracy	Precision	Recall
CNN (word)	TripAdvisor	0.86	0.91	0.82
	Yelp	**0.95**	0.93	**0.97**
CNN (character)	TripAdvisor	**0.90**	**0.92**	**0.88**
	Yelp	0.92	**0.94**	0.90
LSTM + CNN (character)	TripAdvisor	0.87	0.88	0.86
	Yelp	0.93	0.92	0.93
CNN + LSTM (character)	TripAdvisor	0.80	0.82	0.78
	Yelp	0.92	0.92	0.91
RNN Attention (word)	TripAdvisor	0.87	0.88	0.84
	Yelp	0.93	0.92	0.94
RNN Attention + CNN (word)	TripAdvisor	0.86	0.87	0.86
	Yelp	0.94	**0.94**	0.94
Ott et al. [14]	TripAdvisor	0.88	0.88	0.89
Mukherjee et al. [6]	Yelp	0.85	0.87	0.83

pre-trained embedding vectors to initialize unknown words. The character embeddings are initialized with uniform distribution from $[-1, 1]$. The dimension of character embeddings is set to 16. During the training process, character embeddings are fine-tuned.

Results: Table 4 lists the classification performance under different neural network models, including CNN using word embeddings and character embeddings, combination of CNN and LSTM, and RNN with the attention mechanism.

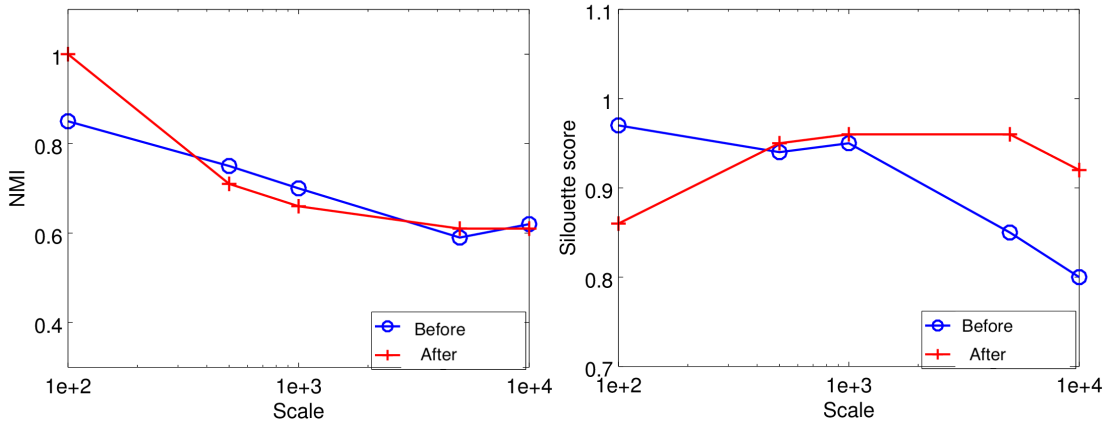
We select CNN using character embeddings as the model for generating deceptive view, due to its simplicity and high overall performance. A combination of Yelp and TripAdvisor datasets is utilized to train the classifier, which can achieve accuracy of 0.93, precision of 0.92, and recall of 0.94. The excellent classification result could imply that the neural network model we choose can well capture the deception features of deceptive reviews. Specifically, we take the fully connected layer of CNN as the the deceptive representation of reviews.

3.2. Integrating Deceptive Views

We integrate deceptive views generated by neural network models into the multiview clustering method. Figure 9 shows the result. After integrating deceptive views, clustering can achieve a higher silhouette score, especially when the reviewer scale is large. However, integrating deceptive views has little impact upon NMI, as NMI remains nearly the same. Overall, integrating deceptive views can improve the homogeneity of clusterings, which means that the reviewers in the same clusterings are more similar to each other.

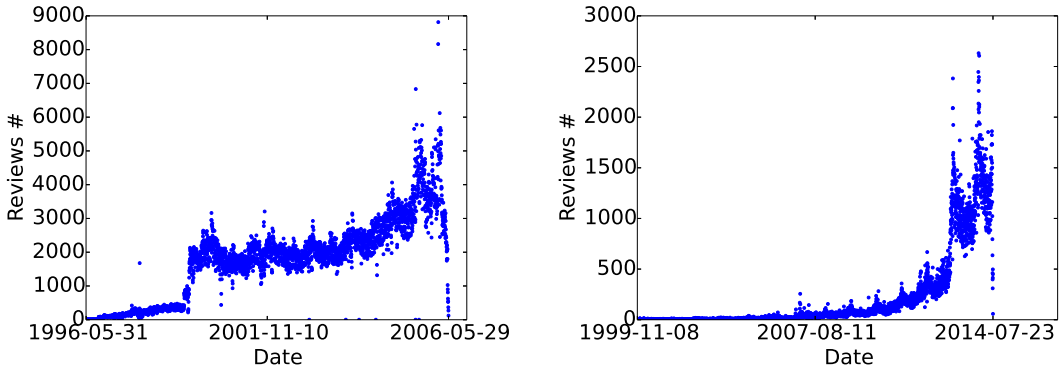
4. A Case Study

We implement the multiview clustering framework and evaluate its capability of detecting deceptive reviews associated with deceptive review writers using large-scale datasets.



(a) NMI. (b) Silhouette score.

Fig. 9. Before vs. after integrating deceptive views.



(a) Dataset 1. (b) Dataset 2.

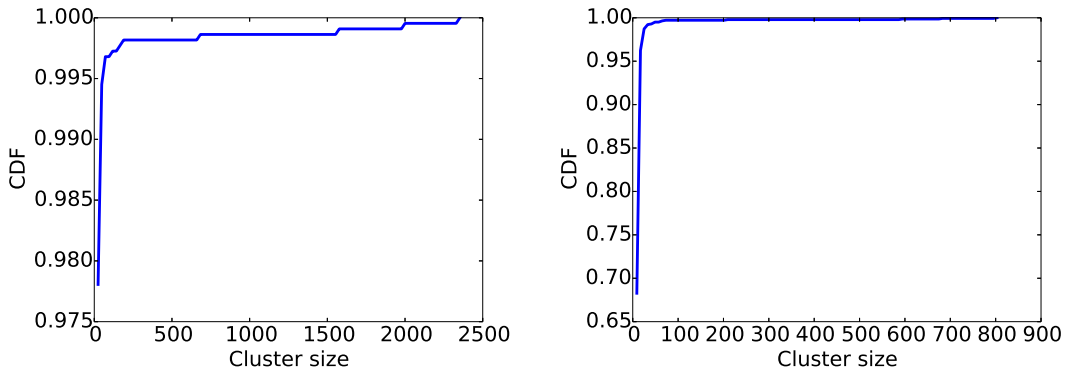
Fig. 10. Dataset description.

4.1. Experiment Setup

We perform the case study with two datasets on a PowerEdge T630 server with 48 of Intel Xeon E5-2690 v3 2.60GHz, 30M Cache CPUs and 4 of 32GB memory. We utilize two Amazon review datasets (*i.e.*, dataset 1 [7] and dataset 2 [8]) to detect deceptive reviewers. Dataset 1 contains all categories of products. However, in dataset 2, we only select four categories of products, including Home Improvement, Home and Kitchen, Toys and Game, and Beauty.

4.2. Dataset Description

Figure 10 shows review count against time of datasets. Dataset 1 ranges from May 31, 1996 to May 29, 2006, and dataset 2 ranges from June 1, 2006 to July 23, 2014.



(a) Dataset 1. (b) Dataset 2.

Fig. 11. Cluster size distribution.

4.3. Cluster Size Distribution

Figure 11 depicts the CDF distribution of cluster size after conducting clustering analysis on two datasets. For both datasets, most of clusters are smaller than 10, while a few of them have extremely large size, especially in dataset 1.

4.4. Preliminary Author Selection

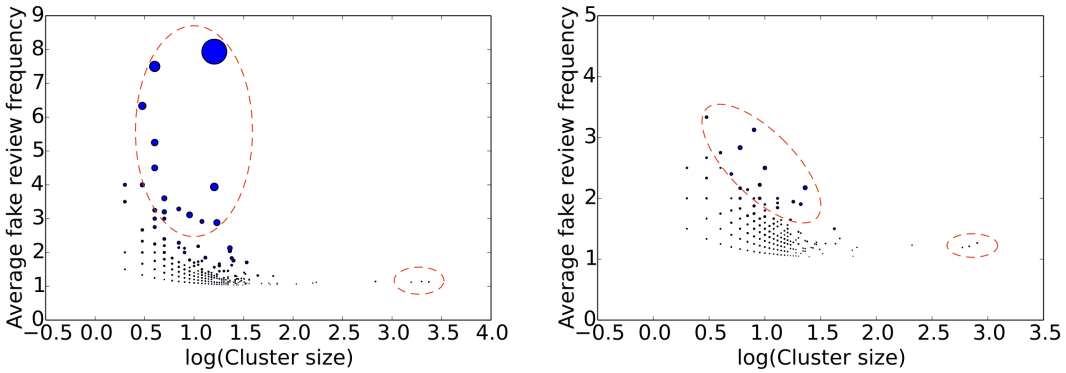
Since the matrix calculation of similarity networks consumes quite an amount of memory resource, the easiest way of conducting large scale detection is to increase memory capacity given the memory is not very expensive. However, an alternative method is to focus on suspicious reviewers, and thus we can use a much less memory capacity to achieve large scale detection. We conduct a preliminary reviewer selection to focus on the suspicious reviewers and reduce the scale. The procedure of the preliminary author selection is performed as follows:

- **Step** 1**:** Select P reviewers who have high deceptive potential according to the deceptive review classifier mentioned above.
- **Step** 2**:** Find K nearest neighbors for each of the above P authors based on the author affinity matrix obtained from authorship and neural network features. Overall, we choose $P \times K$ authors via the preliminary selection.

We set $K = 2$ to obtain the preliminary author set. The size is about $20,000$ and $14,000$ for dataset 1 and dataset 2, respectively. Note that we can set larger K to have a larger preliminary author set. However, due to the memory limitation of our server, we only consider the case of $K = 2$.

4.5. Clustering Results

We obtain the cluster number by searching the optimal cluster number from $1,000$ to $2,500$ according to silhouette score. For dataset 1, we find $1,037$ clusters. The silhouette score is 0.71. For dataset 2, we find $1,409$ clusters. The silhouette score is 0.93, which indicates that the reviewers within same



(a) Dataset 1. (b) Dataset 2.

Fig. 12. Average deceptive review frequency vs. cluster size. The dotted red line indicates clusters that are outliers compared to the rest of the reviews.

cluster are considerably homogeneous with respect to the writing style. The reasons why the silhouette score of dataset 2 is higher than that of dataset 1 are as follows. (1) Dataset 1 contains all categories of products, while dataset 2 only contains four categories. (2) Dataset 1 has more reviewers than dataset 2. Therefore, in some situations, we can lessen the number of categories or the scale of reviewers to improve the performance of clustering.

4.5.1. Average Deceptive Review Frequency

We refer to the reviews that are classified as being deceptive by the neural network model as *deceptive reviews*. Deceptive review frequency can be used to indicate whether a reviewer is deceptive review writer or not. For each cluster, we compute the average deceptive review frequency over reviewers within the cluster. Figure 12 shows the average deceptive review frequency against the cluster size. The circle radius represents the standard variation of reviewers' deceptive frequency within the cluster. For both datasets, we observe two interesting areas. One is in the left dashed oval, which covers the clusters that have considerably high deceptive review frequency, referred to as *HDRF clusters*. This area of our interest represents the clusters with extremely high fake review frequency and moderate cluster size. The other is in the right dashed oval, which covers the clusters of large size, referred to as *LS clusters*. They represent the clusters with low fake review frequency but extremely large cluster size.

To explore these two areas, we select 10 clusters from the area of high deceptive review frequency and three clusters from the area of large size, referred to as *deceptive clusters* and *large clusters*, respectively.

4.5.2. Reviews on Same Products

An ultimate objective of deceptive reviews is to upgrade or downgrade the product reputation. We investigate the number of reviews on the same products within a cluster. Table 5 lists the average number of reviews on the same products for HDRF clusters, LS clusters, and other clusters. For both datasets 1 and 2, HDRF clusters have much higher average number of reviews than the other clusters. For LS and other clusters, the average number of reviews on same products in dataset 1 is much smaller than that in dataset 2.

Table 5
Average number of reviews on a same product.

	HDRF clusters	LS clusters	Other clusters
Dataset 1	1,605	237	243
Dataset 2	1,232	1,035	788

Table 6
Average reviewer lifespan.

	HDRF clusters	LS clusters	Other clusters
Dataset 1	667	386	491
Dataset 2	823	826	850

4.5.3. Reviewer Lifespan

We compute the lifespan of a reviewer, which ranges from the first review to the last review. The lifespan of review accounts could reveal the strategy of deceptive review writers. Table 6 lists the average reviewer lifespan for HDRF clusters, LS clusters, and other clusters. For dataset 1, in terms of the average reviewer lifespan, HDRF clusters are higher than LS and other clusters. However, for dataset 2, they are quite close to each other. It is hard to draw any conclusion from the results. We further inspect the average ratio of one-day lifespan for HDRF clusters, LS clusters, and other clusters, and the results are listed in Table 7. The one-day lifespan means that an account only lives for one day. We observe that HDRF clusters have much more reviewers with one-day lifespan than other clusters for both datasets, and LS clusters behave very differently in the two datasets. It is probably because the number of LS clusters is quite small and the size of each LS cluster is considerably large.

Table 7
Average ratio of one-day lifespan.

	HDRF clusters	LS clusters	Other clusters
Dataset 1	0.55%	11.11%	0.01%
Dataset 2	0.45%	0.04%	0.008%

4.5.4. Average Helpful Feedback

Table 8 lists the average number of helpful feedbacks for HDRF clusters, LS clusters, and other clusters. In dataset 1, HDRF clusters have the highest average number of helpful feedbacks. In dataset 2, HDRF clusters are also close to the highest (other clusters). Thus, we can see that HDRF clusters are effective to deceive human users.

Table 8
Average helpful feedback.

	HDRF clusters	LS clusters	Other clusters
Dataset 1	7.0	5.5	5.0
Dataset 2	2.3	0.8	2.8

4.5.5. Reviewer Ranking

Figure 13 illustrates the CDF of reviewer ranking for HDRF clusters, LS clusters, and other clusters. We observe that the reviewers in HDRF clusters have slightly lower ranking than those in LS clusters

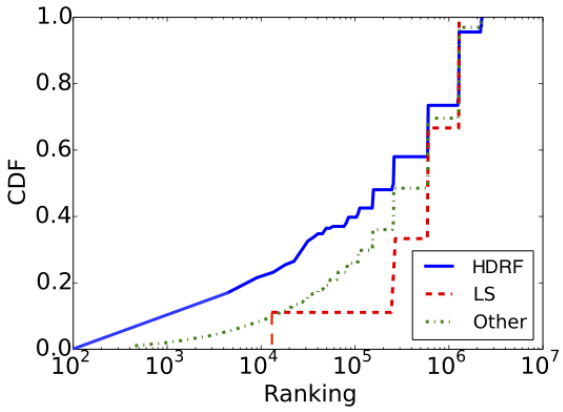


Fig. 13. CDF of reviewer ranking.

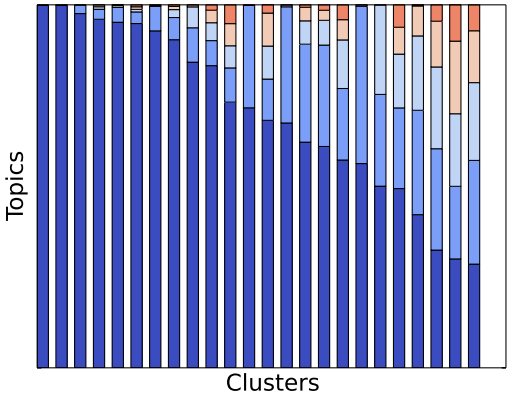


Fig. 14. Topic distribution of HDRF clusters.

and other clusters. The lower ranking a reviewer has, the more reputable he/she is. Some reviewers in HDRF clusters have higher reputation because they are more active, *e.g.*, posting more reviews.

4.5.6. Review Content of HDRF clusters

We examine the content of reviews in HDRF clusters. First, we observe some duplicate reviews in most of HDRF clusters. Note that we can easily detect duplicate or similar content by leveraging writing style features. Second, we utilize Scikit Learn [16] to model the review topics in HDRF clusters. We perform topic modeling by using Latent Dirichlet Allocation (LDA), and we extract five topics from each cluster. Figure 14 depicts the topic distribution across HDRF clusters. We can see that within most clusters, there exists a dominant topic and even almost all the reviews within a cluster are associated with the same topic.

4.5.7. Summary

Among the output clusters of our clustering framework, we focus on HDRF clusters and examine these clusters from different aspects. We reveal that these clusters have the following unusual characteristics:

- HDRF clusters possess much more suspicious reviews that are classified as deceptive, including duplicate reviews, similar reviews, and other deceptive reviews.
- HDRF clusters have much more reviewers who live for only one day, even though a few reviewers within HDRF clusters are quite active and have high reputation.
- Most of HDRF clusters center on only few topics (*e.g.*, a product or one kind of products on Amazon). Given the finding that HDRF clusters have more suspiciously deceptive reviews, we believe that the reviewers within the same HDRF cluster possess the same objective of promoting a certain product or a certain kind of products.

5. Limitation and Future Work

Despite the challenge of limited ground truth, our clustering method is capable of highlighting the suspicious reviewer clusters, as our case study shows. Our clustering result can provide a much-narrowed-down suspect list for further investigation. In our future work, we plan to increase additional views of reviewers to further improve the detection accuracy and detect fake news by leveraging the similarity between deceptive reviews and fake news.

Additional Views of Reviewers: In this work, we only use writing style to detect deceptive reviews. Under the multiview clustering framework, additional views can also be taken into account to improve the detection accuracy, such as a reviewer's profile, a reviewer's history, and product information. For example, a group of reviewers who always publish reviews together are likely to be controlled by the same operator. The aggregation of reviewers' histories can reveal the collusion.

Fake News Detection: The massive spread of fake news in Online Social Networks (OSNs) has been identified as a major global risk, which is alleged to influence election and hurt democracy. Some fake news are also written periodically by a few writers via multiple pseudonyms. It is likely that our detection method could be applied to detect this kind of fake news. News is usually longer than reviews and hence has more abundant corpus for studying authorship attribution and training neural network classifiers.

6. Related Works

Spam Review Detection: Yao et al. [17] presented a potential attack against online review systems by employing deep learning to automatically generate fake reviews. They also proposed countermeasures against these fake reviews. Xie et al. [18] utilized temporal patterns to detect singleton review spam. Wang et al. [19] built review graphs to capture the relationships among reviewers, reviews, and stores, and then quantified the trustiness of reviewers. Zheng et al. [20] attempted to detect elite sybil fake reviews in sybil campaigns. Mukherjee et al. [7] identified fake reviewer groups via behavioral features. Rayana et al. [21] exploited behavioral data, text data, and relational data to detect spam reviews and reviewers. Ott et al. [5, 14] detected deceptive reviews from both positive and negative sentiment review datasets. Song et al. [22] investigated syntactic stylometry for deception detection. Li et al. [23] detected deceptive opinion spam across different domains. Mukherjee et al. [6] examined filtered reviews of YELP and inferred their filtering algorithms. Fusilier et al. [24] employed character n-gram features to detect deceptive opinion spam. Harris et al. [1] examined a variety of human-based, machine-based, and hybrid assessment methods to detect deceptive opinion spam in product reviews. Xu et al. [25] developed a review publication mechanism for local business service systems. Li et al. [26] introduced a motif clustering method for graph analysis. Zhu et al. [27] proposed a information fusion-based framework to

extract the useful information for spam review detection. Fei et al. [28] proposed a detection method for changed-hands accounts used by spammers for a cover of spamming activities.

Reputation Manipulation: In online markets, sellers' reputation is closely related to the profitability. Dishonest sellers have been reported to manipulate the reputation system by faking the transaction history. Xu et al. [29] investigated the underground market by which sellers could easily harness human labors to conduct fake transactions for improving their stores' reputation. They referred to this underground market as Seller-Reputation-Escalation (SRE) markets. Cai et al. [30] employed reinforcement learning methods to detect the reputation manipulation in online markets. Ming et al. [31] implemented deep-learning based method to detect online conterfeit-seller.

In addition, Xie et al. [32] inspected the underground market where mobile app developers could misuse positive reviews illegally. They also analyzed the promotion incentives and characteristics of promoted apps and suspicious reviews. In [33], the authors exploited the unusual ranking change patterns of apps to identify promoted apps and detected the collusive groups who posted high app ratings or inflated apps' downloads.

Authorship Attribution: Shrestha et al. [34] investigated the authorship attribution of online reviews. They employed extensive features (*e.g.*, lexical and syntactic features) to train a verifier for each single reviewer. Two datasets were used to evaluate the verifiers. They attempted to predict the author of a single review or a collection of reviews. In [35], the authors studied the authorship attribution of reviews by transforming document features into similarity features. Using the similarity features, they leveraged learning algorithms to build the classifiers.

In [36] the authors surveyed stylometry in different objectives while [37] used stylometry to identify the authors in the underground forums. Hitschler et al. [38] used a convolutional neural network to perform authorship identification on a dataset of scientific publications. They only utilized stylometric features. Koppel et al. [39] reasoned that any authorship attribution problem could be broken into a set of authorship verification problems. They then converted the verification problem into many-candidates problem by generating a large set of impostor candidates. If two documents are consistently more similar to each other than to the other generated impostors, they are probable to be written by the same author. Narayanan et al. [12] investigated the possibility of identifying authors of anonymously published content based on stylometry techniques.

Deep Neural Networks in Text Classification: Kalchbrenner et al. [40] adopted a dynamic convolutional neural network for the semantic modeling of sentences. Kim [41] studied sentence classification by using convolutional neural networks built on top of word2vec, and found that a simple convolutional neural network with one layer of convolution performs remarkably well. Zhang et al. [42] explored the use of character-level convolutional networks for text classification. Dong et al. [43] built a hierarchical sentence-document model to perform automatic essay scoring. In [11], the authors explored a neural network model to learn document-level representation for detecting deceptive opinion spam. Zhou et al. [44] explored deep learning method for user identity linkage. Zhao et al. [45] developed a deep learning framework for friendship inference in online social networks. Xiang et al. [46] employed deep learning method for protecting privacy from optimization perspective.

7. Conclusions

We proposed a novel approach to detecting deceptive reviews written by professional review writers. We leveraged authorship attribution to identify the writing style of reviewers, and employed a multiview

clustering method to group authors with similar writing style. In addition, we compared different neural network models for characterizing deceptive writing styles. We compared the performance of different classifiers, showing that a convolutional neural network has the best overall detection accuracy of 90%. Finally, we evaluated the effectiveness of the multiview clustering framework based on large scale Amazon datasets as a case study and we demonstrated that clustering method outperforms existing K-means and hierarchical methods.

References

[1] C.G. Harris, Detecting Deceptive Opinion Spam Using Human Computation, *Workshops at the 26th AAAI Conference on Artificial Intelligence* (2012), 87–93. ISBN 9781577355731.

[2] J. Jia, B. Wang and N.Z. Gong, Random Walk Based Fake Account Detection in Online Social Networks, in: *47th Annual IEEE/IFIP International Conference on Dependable Systems and Networks*, 2017, pp. 273–284. ISSN 2158-3927. doi:10.1109/DSN.2017.55.

[3] J. Jin, J. Offutt, N. Zheng, F. Mao, A. Koehl and H. Wang, Evasive Bots Masquerading as Human Beings on the Web, in: *43rd Annual IEEE/IFIP International Conference on Dependable Systems and Networks*, 2013, pp. 1–12.

[4] X. Ruan, Z. Wu, H. Wang and S. Jajodia, Profiling Online Social Behaviors for Compromised Account Detection, *IEEE Transactions on Information Forensics and Security* **11**(1) (2016), 176–187. doi:10.1109/TIFS.2015.2482465.

[5] M. Ott, Y. Choi, C. Cardie and J.T. Hancock, Finding Deceptive Opinion Spam by Any Stretch of the Imagination, in: *NAACL-HLT*, 2011, pp. 309–319. ISSN 19909772. ISBN 978-1-932432-87-9. doi:10.1145/2567948.2577293.

[6] A. Mukherjee, V. Venkataraman, B. Liu and N.S. Glance, What yelp fake review filter might be doing?, in: *ICWSM*, 2013, pp. 409–418.

[7] A. Mukherjee, B. Liu and N. Glance, Spotting Fake Reviewer Groups in Consumer Reviews, in: *International Conference on World Wide Web (WWW)*, 2012, p. 191. ISSN 0025-1909. ISBN 9781450312295. doi:10.1145/2187836.2187863.

[8] R. He and J. McAuley, Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering, in: *Proceedings of the 25th International Conference on World Wide Web (WWW)*, 2016, pp. 507–517.

[9] Y. Zhang, X. Ruan, H. Wang and H. Wang, What scale of audience a campaign can reach in what price on twitter?, in: *IEEE INFOCOM 2014-IEEE Conference on Computer Communications*, IEEE, 2014, pp. 1168–1176.

[10] B. Wang, J. Jiang, W. Wang, Z.H. Zhou and Z. Tu, Unsupervised metric fusion by cross diffusion, *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition* (2012), 2997–3004. ISBN 9781467312264. doi:10.1109/CVPR.2012.6248029.

[11] Y. Ren and Y. Zhang, Deceptive Opinion Spam Detection Using Neural Network, in: *Proceedings of the 26th International Conference on Computational Linguistics*, 2016, pp. 140–150.

[12] A. Narayanan, H. Paskov, N.Z. Gong, J. Bethencourt, E. Stefanov, E.C.R. Shin and D. Song, On the feasibility of internet-scale author identification, in: *IEEE Symposium on Security and Privacy*, 2012, pp. 300–314. ISSN 10816011. ISBN 9780769546810. doi:10.1109/SP.2012.46.

[13] Clustering.

[14] M. Ott, C. Cardie and J.T. Hancock, Negative Deceptive Opinion Spam, in: *NAACL-HLT*, 2013, pp. 497–501. ISBN 9781937284473.

[15] J. Pennington, R. Socher and C. Manning, Glove: Global Vectors for Word Representation, in: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2014, pp. 1532–1543. ISSN 10495258. ISBN 9781937284961. doi:10.3115/v1/D14-1162.

[16] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel and B. Thirion, Scikit-learn: Machine Learning in Python Gaël Varoquaux, *Journal of Machine Learning Research* **12** (2011), 2825–2830.

[17] Y. Yao, B. Viswanath, J. Cryan, H. Zheng and B.Y. Zhao, Automated Crowdturfing Attacks and Defenses in Online Review Systems, in: *CCS*, 2017. ISBN 9781450349468. doi:10.1145/3133956.3133990.

[18] S. Xie, G. Wang, S. Lin and P.S. Yu, Review Spam Detection via Temporal Pattern Discovery, in: *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2012, pp. 823–831. ISBN 9781450314626. doi:10.1145/2339530.2339662.

[19] G. Wang, S. Xie, B. Liu and P.S. Yu, Review Graph Based Online Store Review Spammer Detection, in: *IEEE International Conference on Data Mining*, 2011, pp. 1242–1247. ISSN 15504786. ISBN 9780769544083. doi:10.1109/ICDM.2011.124.

[20] H. Zheng, M. Xue, H. Lu, S. Hao, H. Zhu, X. Liang and K. Ross, Smoke Screener or Straight Shooter: Detecting Elite Sybil Attacks in User-Review Social Networks, in: *NDSS*, 2018. doi:10.14722/ndss.2018.23009.

[21] S. Rayana and L. Akoglu, Collective Opinion Spam Detection: Bridging Review Networks and Metadata, in: *Proceedings of the 21th ACM International Conference on Knowledge Discovery and Data Mining*, 2015. doi:10.1145/2783258.2783370.
[22] S. Feng, R. Banerjee and Y. Choi, Syntactic stylometry for deception detection, *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics* (2012), 171–175. ISBN 9781937284251. doi:10.1017/CBO9781107415324.004.
[23] J. Li, M. Ott, C. Cardie and E. Hovy, Towards a General Rule for Identifying Deceptive Opinion Spam, *ACL* (2014), 1566–1576. ISBN 9781937284725.
[24] D.H. Fusilier, M. Montes-Y-Gomez, P. Rosso and R.G. Cabrera, Detection of opinion spam with character n-grams, in: *Lecture Notes in Computer Science*, Vol. 9042, Springer International Publishing, 2015, pp. 285–294. ISSN 16113349. ISBN 9783319181165. doi:10.1007/978-3-319-18117-2$_2$1.
[25] X. Zheng, Z. Cai, J. Li and H. Gao, Location-privacy-aware review publication mechanism for local business service systems, in: *IEEE INFOCOM 2017-IEEE Conference on Computer Communications*, IEEE, 2017, pp. 1–9.
[26] P. Li, H. Dau, G. Puleo and O. Milenkovic, Motif clustering and overlapping clustering for social network analysis, in: *IEEE INFOCOM 2017-IEEE Conference on Computer Communications*, IEEE, 2017, pp. 1–9.
[27] Y. Zhu, H. Liu, Y. Du and Z. Wu, IFSpard: An Information Fusion-based Framework for Spam Review Detection, in: *Proceedings of the Web Conference 2021*, 2021, pp. 507–517.
[28] G. Fei, S. Wang, B. Liu and L. Akoglu, Detecting Changed-Hands Online Review Accounts, *arXiv preprint arXiv:2106.15352* (2021).
[29] H. Xu, D. Liu, H. Wang and A. Stavrou, E-commerce Reputation Manipulation: The Emergence of Reputation-Escalation-As-A-Service, in: *International Conference on World Wide Web (WWW)*, 2015, pp. 1296–1306.
[30] Q. Cai, A. Filos-Ratsikas, P. Tang and Y. Zhang, Reinforcement Mechanism Design for Fraudulent Behaviour in E-commerce, in: *Proceedings of the 32nd AAAI Conference on Artificial Intelligence*, 2018.
[31] M. Cheung, J. She and L. Liu, Deep learning-based online counterfeit-seller detection, in: *IEEE INFOCOM 2018-IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, IEEE, 2018, pp. 51–56.
[32] Z. Xie and S. Zhu, AppWatcher: Unveiling the Underground Market of Trading Mobile App Reviews, in: *Proceedings of the 8th ACM Conference on Security & Privacy in Wireless and Mobile Networks*, 2015, p. 10.
[33] H. Chen, D. He, S. Zhu and J. Yang, Toward Detecting Collusive Ranking Manipulation Attackers in Mobile App Markets, in: *Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security*, 2017, pp. 58–70.
[34] P. Shrestha, A. Mukherjee and T. Solorio, Large Scale Authorship Attribution of Online Reviews, *Conference on Intelligent Text Processing and Computational Linguistics* (2016).
[35] T.Y. Qian, B. Liu, Q. Li and J. Si, Review Authorship Attribution in a Similarity Space, *Journal of Computer Science and Technology* **30**(1) (2015), 200–213. ISBN 1139001515. doi:10.1007/s11390-015-1513-6.
[36] T. Neal, K. Sundararajan, A. Fatima, Y. Yan, Y. Xiang and D. Woodard, Surveying stylometry techniques and applications, *ACM Computing Surveys (CSUR)* **50**(6) (2017), 1–36.
[37] S. Afroz, A.C. Islam, A. Stolerman, R. Greenstadt and D. McCoy, Doppelgänger finder: Taking stylometry to the underground, in: *2014 IEEE Symposium on Security and Privacy*, IEEE, 2014, pp. 212–226.
[38] J. Hitschler, E.V.D. Berg and I. Rehbein, Authorship Attribution with Convolutional Neural Networks and, *Association for Computational Linguistics* (2017), 53–58.
[39] M. Koppel and Y. Winter, Determining If Two Documents Are Written By the Same Author, *Journal of the American Society for Information Science and Technology* **65**(1) (2014), 178–187. ISBN 9783848215430. doi:10.1002/asi.22954.
[40] N. Kalchbrenner, E. Grefenstette and P. Blunsom, A Convolutional Neural Network for Modelling Sentences, in: *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, 2014, pp. 655–665. ISBN 9781937284725. doi:10.3115/v1/P14-1062.
[41] Y. Kim, Convolutional Neural Networks for Sentence Classification, *CoRR* (2014). ISBN 9781937284961. doi:10.3115/v1/D14-1181.
[42] X. Zhang, J. Zhao and Y. LeCun, Character-level convolutional networks for text classification, in: *Advances in Neural Information Processing Systems*, 2015.
[43] F. Dong, Y. Zhang and J. Yang, Attention-based Recurrent Convolutional Neural Network for Automatic Essay Scoring, in: *Proceedings of the 21st Conference on Computational Natural Language Learning*, 2017, pp. 153–162.
[44] F. Zhou, L. Liu, K. Zhang, G. Trajcevski, J. Wu and T. Zhong, Deeplink: A deep learning approach for user identity linkage, in: *IEEE INFOCOM 2018-IEEE Conference on Computer Communications*, IEEE, 2018, pp. 1313–1321.
[45] Y. Zhao, M. Qiao, H. Wang, R. Zhang, D. Wang, K. Xu and Q. Tan, TDFI: Two-stage Deep Learning Framework for Friendship Inference via Multi-source Information, in: *IEEE INFOCOM 2019-IEEE Conference on Computer Communications*, IEEE, 2019, pp. 1981–1989.
[46] L. Xiang, J. Yang and B. Li, Differentially-Private Deep Learning from an optimization Perspective, in: *IEEE INFOCOM 2019-IEEE Conference on Computer Communications*, IEEE, 2019, pp. 559–567.