Synthetic Review Spamming and Defense

Huan Sun, Alex Morales, and Xifeng Yan
Department of Computer Science
University of California, Santa Barbara
{huansun, alex_morales, xyan}@cs.ucsb.edu

ABSTRACT

Online reviews have been popularly adopted in many applications. Since they can either promote or harm the reputation of a product or a service, buying and selling fake reviews becomes a profitable business and a big threat. In this paper, we introduce a very simple, but powerful review spamming technique that could fail the existing feature-based detection algorithms easily. It uses one truthful review as a template, and replaces its sentences with those from other reviews in a repository. Fake reviews generated by this mechanism are extremely hard to detect: Both the state-of-the-art computational approaches and human readers acquire an error rate of 35%-48%, just slightly better than a random guess. While it is challenging to detect such fake reviews, we have made solid progress in suppressing them. A novel defense method that leverages the difference of semantic flows between synthetic and truthful reviews is developed, which is able to reduce the detection error rate to approximately 22%, a significant improvement over the performance of existing approaches. Nevertheless, it is still a challenging research task to further decrease the error rate.

Synthetic Review Spamming Demo:
`www.cs.ucsb.edu/~alex_morales/reviewspam/`

Categories and Subject Descriptors

I.2.7 [**Artificial Intelligence**]: Natural Language Processing—*text analysis*; K.4.1 [**Computers and Society**]: Public Policy Issues—*abuse and crime involving computers*

Keywords

Review Spam; Spam Detection; Classification

1. INTRODUCTION

Online reviews are widely adopted in many websites such as Amazon, Yelp, and TripAdvisor, allowing users to exchange their personal experiences. Positive reviews can increase reputations and bring significant financial gains, while negative ones often cause dramatic sales loss. This fact, unfortunately, results in strong incentives for review spam, *i.e.*, writing fake reviews to mislead readers.

(***a***) I recently stayed at the Bryant Park Hotel and was happy with every single aspect of it: great location; friendly staff; beautiful room and a picture postcard view from our window. Very contemporary rooms are comfortable and spacious. The Cellar Bar was is a very trendy bar located at the hotel that seemed very popular. Our room looked out on Bryant Park, so we had a nightly view of the ice skating rink and holiday lights. I will definitely stay here again, having tried a dozen or so other NYC hotels.

(***b***) My husband and I stayed at the James Chicago Hotel for our anniversary. This place is fantastic! We knew as soon as we arrived we made the right choice! The rooms are BEAUTIFUL and the staff very attentive and wonderful!! The area of the hotel is great, since I love to shop I couldn't ask for more!! We will definitely back to Chicago and we will for sure be back to the James Chicago.

(***c***) This is best hotel bargain in Chicago if you are not overly unhappy with a small room. Great hotel just redone by Kimpton. It is one of the best that I have stayed in. This is a location just off of LaSalle and next to the Cadillac theater. The Kimptons have a social hour 5 to 6 with wine at all of their hotels. For a total of $100. They are super animal friendly and provide complete services to your dog if you wish to bring it. The staff from top to bottom were excellent, accommodating. Just walking in, the hotel was gorgeous. Always consider Kimptons as they never disappoint where ever we have used them.

Figure 1: Truthful or Spam Reviews

To show how challenging spam detection is, we first illustrate three example reviews in Figure 1, among which two are fake reviews. Without very careful scrutiny, it is even difficult for human readers to identify deceptive reviews from truthful ones. Review (a) is a truthful review; (b) is a deceptive review written by a customer who hasn't visited that hotel; and (c) is a deceptive review synthesized from multiple truthful reviews (our algorithm).

Several algorithms have been proposed to suppress the growth of review spam. For instance, Jindal *et al.* [11] focused on the detection of disruptive review spam (e.g., comments on brands only, non-opinion texts), which is less threatening due to easy identification by human readers. Feng *et al.* [6] proposed a detection strategy based on the assumption that fake reviews tend to distort the natural distribution of review scores. However, these techniques are not effective in handling the new kinds of fake reviews shown in Figure 1, since they are indeed authentic-looking reviews either deliberately written by humans or synthesized from other human writings.

Ott *et al.* [19] studied deceptive reviews (e.g., (b) in Figure 1) that are deliberately written to mislead readers. Based on n-gram and psychological deception features, the detector proposed by Ott *et al.* can achieve nearly 90% accuracy for human written deceptive reviews. It is observed that liars tend to have different writing patterns, such as absence of specific spatial information and excessive use of exclamatory marks, from reviewers that have true experiences [19].

Apart from Ott's algorithm's good performance, one has to spend money to hire human writers, in order to generate such deceptive reviews. Actually, Ott *et al.* [19] spent $1 per review to create 400 deceptive reviews using Amazon Mechanical Turkers in 14 days. In this paper, we first pretend to be evil by asking the following question *"If we were attackers, can we employ a much more economical, high-throughput approach to generate deceptive reviews with minimum human involvement?"*. Setting off from this malicious perspective, we bring into attention an automatic review synthesis process. We show that an attacker is able to fake an authentic-looking review (such as (c) in Figure 1) by merely organizing sentences extracted from existing online truthful reviews. This process works in a much more economical way in terms of both time and costs; and the most thrilling result is that the state-of-the-art detection algorithms including Ott's are not able to handle this kind of fake reviews: The detection error rate is around 35%-44%, just slightly better than a random guess.

The reason behind the failure of detecting the machine-synthesized reviews is quite simple: All the sentences in these reviews were actually written by people who *had* true experiences. To counter such spam attacks, we develop a new defense framework, based on the observation that subtle semantic incoherence exists in a synthesized review. Pairwise and multiple sentence coherence features are developed to improve the detection performance. Classifiers built on these features can reduce the detection error rate to around 22%, a significant improvement over the existing approaches.

To summarize, our contributions are two-fold: (1) We bring into attention an automated, low-cost process for generating fake reviews, variations of which could be easily employed by evil attackers in reality. To the best of our knowledge, we are the first to expose the potential risk of machine-generated deceptive reviews. By doing that, we aim at stimulating debate and defense against this deception scheme; (2) Incapable are the state-of-the-art detection algorithms which only deal with disruptive spam and human-written deceptive reviews. We fill the hole by proposing a general framework to detect machine-synthesized fake reviews, and further instantiate this framework with our proposed measures that capture semantic coherence and flow smoothness.

The rest of the paper is organized as follows. In Section 2, we describe an automatic process to synthesize deceptive reviews and show how current deception detectors and human readers perform on synthetic reviews. Based on coherence examination, we propose a general framework to detect synthetic reviews in Section 3, followed by instantiations of the framework in Section 4. Section 5 presents our detailed experimental results. Related work is reviewed in Section 6. We conclude this work in Section 7.

2. AUTOMATED REVIEW GENERATION

In this section, we demonstrate a review synthesis method that is able to generate deceptive reviews automatically from a pool of truthful reviews, followed by performance analysis of human readers and existing detection algorithms.

2.1 A Review Synthesis Model

Can we develop a model to automatically generate positive reviews by mixing up those existing reviews? Not only it is possible (for the same category of products or services), it works perfectly to fool human readers and detection algorithms. We now describe this process shown in Figure 2.

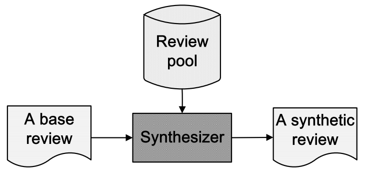

Figure 2: Review Synthesization

[***Review pool***] Take hotel review as an example. We first collect truthful reviews from online websites like TripAdvisor with high positive scores and containing more than 150 characters. We discard short reviews since many of them are not content rich.
[***Base review***] A base review is randomly drawn from the pool, based on which a synthetic review will be generated.
[***Synthesizer***] A synthesizer takes the base review as a template and synthesize a new one using the reviews in the pool. There could be many mechanisms to transform a base review to a synthetic review using different rewriting techniques. In our work, we adopt the simplest strategy: The synthesizer replaces each sentence in a base review by the most similar (not exactly the same) sentence in the review pool. A synthetic review is output after a full replacement of sentences in the base review. This strategy is simple, but very effective.

Specifically, let R represent our review pool and T denote the base review set. The review synthesizer works as follows:

(1) For a base review $r \in T$ with a sentence sequence $\{s_1, s_2, ..., s_n\}$, get a similar sentence in R for each sentence s_i by

$$s'_i = \arg \max_{s \in R, s \neq s_i} sim(s, s_i),$$

(2) Output a synthesized review r' composed of a sentence sequence $\{s'_1, s'_2, ..., s'_n\}$.
The sim function could be instantiated by any sentence similarity measure. Two typical measures are cosine similarity, and set similarity (the number of overlapped words in two sentences). Let s_i and s_j denote the set of words in two sentences, cosine similarity is defined as $\cos(s_i, s_j) = \frac{v_i^T v_j}{\|v_i\| \|v_j\|}$, where v_i (v_j) is the word frequency vector for sentence s_i (s_j), and $\| \cdot \|$ is the ℓ_2 norm of a vector. Cosine similarity tends to replace one sentence with a shorter one whereas set similarity tends to replace one sentence with a longer one. To make a synthetic review have the similar length of the base review, we randomly choose either cosine similarity or set similarity as sim when doing sentence replacement.

In this way, one cannot use review length as a criterion to detect fake reviews.

A larger review pool tends to help generate more authentic-looking deceptive reviews due to the large variety of sentences. Additionally, in practice one might need to do location/name check in synthesized reviews whereas in this work we put little emphasis on this issue. Our to-be-proposed detection methods will not rely on any location/name information, and besides, human readers are generally able to notice the location/name conflict (if any) alertly.

Intuitively, the above synthesis process can work well due to two reasons. First, the base review, written by human beings, provides a very good skeleton, which makes the synthetic review as authentic-looking as possible. Second, in terms of information flow and content richness, online reviews that consist of at most several paragraphs are much simpler than research papers, news articles, and novels. While automatically synthesizing those complicated texts are much more difficult (though there are successful stories. Some synthetic research papers even passed reviewers' examination [21]), online review synthesis is relatively easier and hard to detect.

2.2 Human Readers

To test human performance on the above synthesized reviews, ten volunteers are solicited. Half of them are graduate students and half are their family members who read online reviews quite often. All of the volunteers do not have knowledge on how fake reviews are generated and detected; they should be good representatives of general customers. The testing dataset contains 10 truthful reviews (from TripAdvisor, we carefully select these reviews) and 10 synthesized reviews. We prefer this small set of reviews because the volunteers can easily get fatigued by many reviews and consequently their performance of detecting fake reviews is reduced. Readers are welcome to try the synthetic review detection task via `www.cs.ucsb.edu/~alex_morales/reviewspam/`.

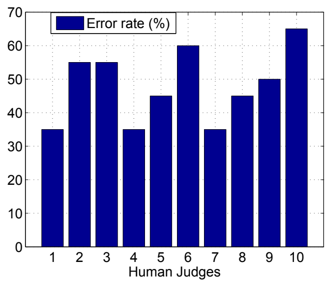

Figure 3: Performance of 10 Human Judges

Figure 3 shows the error rate of human readers, where error rate is defined as the percentage of misclassified reviews over all reviews. The average error rate is around 48%, indicating that it is very hard for a reader to distinguish fake reviews from truthful ones.

2.3 The State-of-the-Art Detection Methods

We also examined the performance of the state-of-the-art fake review detectors [19, 15, 10] on detecting the synthesized reviews. In this set of tests, we used the same experimental setting as in Section 5. Table 1 shows the result of three algorithms.

Algorithms	Error rate (%)
Ott *et al.* [19]	40.5
Liu *et al.* [15]	34.5
Harris *et al.* [10]	43.3

Table 1: Performance of Various Detectors.

The detector developed by Ott *et al.* [19] only obtains 59.5% accuracy when applied to synthesized reviews. It is evident that relying on abnormal writing styles of fake reviews is no longer effective in dealing with synthesized reviews.

Generated by combining sentences from different reviews, the writing of a synthetic review might be of low quality. We therefore also tested a quality-examination method proposed in [15], which extracts statistical features to measure the quality of product reviews. We regard our truthful (synthetic) reviews as high-quality (low-quality) in their setting. This algorithm achieves a 34.5% misclassification error rate.

Harris *et al.* [10] spot deceptive review spam by combining methods in [19] with statistical features extracted from the review text. It achieves a 43.3% error rate.

While the computational approaches outperform human readers, their performance is not impressive. The automated synthesizer, which takes advantage of existing online reviews and simple sentence-replacement strategies, is therefore easy to implement but turns out to generate quite authentic-looking and hardly detectable reviews.

2.4 Discussions

One might generate a synthetic review by simply duplicating the base review. However, a duplication of an entire review could be detected more easily. In contrast, using sentence-wise replacement, one can generate a much larger set of fake reviews, significantly increasing the fake review space and detection difficulty. To pass those sentence-level duplication detectors, one could further use automatic rewriting/paraphrasing techniques [17, 8], e.g., synonym replacement. Generally speaking, the local text content (such as n-grams of a text) is not a unique fingerprint of one specific review. Reviewers might also cite what peers mentioned in previous reviews, or use the same words to praise or criticize. Therefore, using text reuse as an indicator of being fake, e.g., employing the search engine to check whether an n-gram of a review is covered in other texts, might cause high false positive rate. Details involving sentence paraphrase and paraphrase detection will deviate too much from the current focus of this work. We stay focused on the simple sentence replacement strategy and resort to feature-based defense techniques. Nonetheless, our to-be-proposed methodology is directly applicable to paraphrased synthetic reviews.

3. SYNTHETIC REVIEW DETECTION

Compared with natural writings, synthetic reviews using sentence transplants bear subtle semantic incoherence between sentences. We now advocate a general and extensible methodology for coherence analysis, which consists of two components: pairwise sentence coherence and multiple sentence coherence. Figure 4 shows the framework. Each filled circle denotes one sentence in a review. f denotes a general measure (feature) that is imposed on either a sentence pair or multiple sentences.

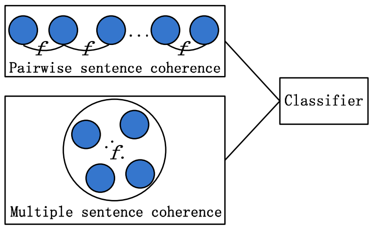

Figure 4: Illustration of Our Methodology.

Now we discuss general perspectives from which we are going to measure the coherence of a review. Each measure will be instantiated with greater details in Section 4. Pairwise sentence coherence aims to measure the information flow smoothness between sentences:

Sentence transition: Due to the coherence nature of human writings, given a word in one sentence, one could expect to observe certain words in its following sentence with some probability.

Word co-occurrence: Similar to the transition property (conditional probability), words generally demonstrate co-occurrence patterns (joint probability) in two consecutive sentences.

Pairwise sentence similarity: Subtly different from the transition and co-occurrence properties, pairwise similarity takes into account the word/semantic overlap between two consecutive sentences.

Multiple sentence coherence measures the stretch and changes of topics in multiple consecutive sentences:

Semantic dispersion: Given a vectorized semantic representation of each sentence (e.g., topic distribution), we quantify how dispersed/focused the content of a review is. Let $\{v_1, v_2, ..., v_n\}$ be the semantic vector representation corresponding to each sentence in one review. We define the semantic dispersion as:

$$SD = \frac{1}{n}\sum_{i=1}^{n} \|v_i - centroid\| \tag{1}$$

where $centroid = \frac{1}{n}\sum_{i=1}^{n} v_i$, and $\|\cdot\|$ refers to the ℓ_2 norm of a vector. Since sentences in a synthetic review are originally from different contexts, we expect SD of synthetic reviews to be generally larger than that of truthful ones.

Running length: Instead of emphasizing the smooth semantic flow, running length takes into consideration the occasional semantic jumps between two adjacent sentences in a review. Given any pairwise sentence similarity measure sim, we compute the similarity between two adjacent sentences s_i and s_{i+1} in a review with a sentence sequence $s_1 \to s_2 \to \cdots \to s_n$. For a given threshold δ, once $sim(s_i, s_{i+1})$ is less than δ, we break the flow edge from s_i to s_{i+1}. The original review is hence segmented to pieces. We count the number of sentences in the k_{th} piece, denoted as l_k and measure the overall compactness of the review by $\sum_{k=1}^{K} l_k/K$, where K is the total number of pieces we have. We denote this measure as *running length* (RL). To make the measure robust, one might consider either choosing similarity functions based on words' semantics or using multiple sentences as a basic unit s_i.

4. CONTENT COHERENCE

In this section, we introduce several instantiations of the aforementioned concepts in Section 3.

4.1 Sentence Transition

We first define one-step transition probability from word w_i to word w_j as the probability of observing w_j by randomly drawing a word in a sentence given word w_i in the previous sentence. The pointwise transition probability matrix (PTP), with each element (i, j), records this probability. Given a set of words W, the transition probabilities from word w_i to other words form the i_{th} row of PTP as follows:

$$\begin{aligned} PTP_{(i,:)} &= [P(w_1|w_i), ..., P(w_j|w_i), ..., P(w_n|w_i)] \\ &s.t. \sum_{w_j \in W} P(w_j|w_i) = 1 \end{aligned}$$

PTP$_{(i,:)}$ can be estimated directly using maximum likelihood estimation from the training dataset. All the consecutive sentence pairs (s, s') are extracted with word w_i in s. The words in s' are observed sample words following w_i. We denote the sentence set formed by s' as S_i. Let $c(w, S_i)$ denote the frequency of w in S_i.

$$P(w_j|w_i) = \frac{c(w_j, S_i)}{\sum_{w_k \in W} c(w_k, S_i)} \tag{2}$$

Based on the pointwise transition probability, for any two sentences s_1 and s_2, the probability to observe s_2 after s_1, i.e. $s_1 \to s_2$, can be estimated using the following different models since s_1 and s_2 have multiple words. Figure 5 shows the pointwise transition probability between two sets of words.

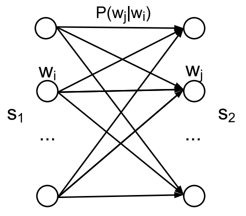

Figure 5: Sentence Transition

Best Edge: Choose the edge with the highest transition probability as the transition probability of two sentences.

$$P^{be}(s_1 \to s_2) = \max_{w_i \in s_1, w_j \in s_2} P(w_j|w_i)^{c(w_j, s_2)} \tag{3}$$

Pivot Node: Choose the node that generates the highest productive transition probability for all the words in s_2.

$$P^{n}(s_1 \to s_2) = \max_{w_i \in s_1} P(|s_2|) \prod_{w_j \in s_2} P(w_j|w_i)^{c(w_j, s_2)} \tag{4}$$

where $|\cdot|$ is the number of words in a sentence. The distribution of sentence length $P(|\cdot|)$ can be estimated from the training set.

Pivot Edge: Choose the best set of edges that generate the highest productive transition probability for all the words in s_2.

$$P^{e}(s_1 \to s_2) = P(|s_2|) \prod_{w_j \in s_2} \max_{w_i \in s_1} P(w_j|w_i)^{c(w_j, s_2)} \tag{5}$$

Mixture: Use a mixture model to mix all the edges together.

$$\begin{aligned} P(w_j|s_1) &= \textstyle\sum_{w_i \in s_1} \theta(w_i, s_1) P(w_j|w_i) \\ P^m(s_1 \to s_2) &= \prod_{w_j \in s_2} P(w_j|s_1)^{c(w_j,s_2)} \end{aligned} \quad (6)$$

where $\theta(w_i, s_1) = c(w_i, s_1)/|s_1|$ weighs word w_i in sentence s_1.

Now we propose a coherence measure of a review based on *perplexity*[2]. Perplexity has been used for the evaluation of different language models whereas in our work we apply it as a feature to discriminate truthful reviews from synthetic reviews. Following the convention, the perplexity of review r with a sentence sequence $\{s_1, s_2, \ldots, s_n\}$ is defined as:

$$\begin{aligned} Perplexity(r) &= \exp\{-\frac{\log(P(s_1 \to s_2 \to \ldots \to s_n))}{\sum_{i=1}^{n-1} |s_{i+1}|}\} \\ &= \exp\{-\frac{\sum_{i=1}^{n-1} \log(P(s_i \to s_{i+1}))}{\sum_{i=1}^{n-1} |s_{i+1}|}\} \end{aligned} \quad (7)$$

We use $\log(Perplexity(r))$ as our PTP coherence measure to make it at the same magnitude with other features proposed later. The measure can be instantiated with the four different sentence transition models proposed above, respectively denoted as ptp^{be}, ptp^{n}, ptp^{e} and ptp^{m}.

The PTP coherence measure works based on the assumption that human writings demonstrate word transition patterns between two consecutive sentences. Such transition patterns can be impaired by sentence replacements in the review synthesis process, in that the new sentences are originally from different reviews. Consequently, the sentence transition models trained from human written reviews generally cannot well explain the observation of two consecutive sentences in a synthetic review (lower likelihood is observed); therefore, larger PTP values on synthetic review are expected, compared with those on truthful reviews.

4.2 Word Co-Occurrence

To capture the word co-occurrence patterns in two consecutive sentences, we leverage three important probabilities: randomly drawing two consecutive sentences, the probability of observing word w_i (denoted as P_i), the probability of observing word w_j (denoted as P_j), and the probability of observing word w_i in one sentence and word w_j in the other (denoted as $P_{i,j}$). All of them are directly estimated from the training dataset. We define the word co-occurrence score for word w_i and w_j as:

$$O_{i,j} = \log(\frac{P_{i,j}}{P_i P_j})$$

This score characterizes the significance of observing two words' co-occurrence in a consecutive sentence pair, in the sense that it not only takes into account the co-occurrence probability, but also counteracts the effect that one word might occur quite frequently by itself and naturally co-occur with many others. The measure differs from the classic Pointwise Mutual Information (PMI) [5] in that $P_{i,j}$ is *not* the joint probability of observing word w_i and w_j in a consecutive sentence pair, where w_i and w_j might co-occur in the same sentence.

Since each sentence contains multiple words, words might co-occur more frequently in long sentences. Therefore, we need to discount the influence of sentence length, either using the average or maximum word co-occurrence score:

Average Score: Use the average of word co-occurrence scores as the sentence co-occurrence score (SCO) for two consecutive sentences.

$$sco^{as}(s_1, s_2) = \frac{1}{|s_1||s_2|} \textstyle\sum_{w_i \in s_1, w_j \in s_2} O_{i,j} \quad (8)$$

Best Score: Choose the highest word co-occurrence score as the sentence co-occurrence score.

$$sco^{bs}(s_1, s_2) = \max_{w_i \in s_1, w_j \in s_2} O_{i,j} \quad (9)$$

For a review r, with a sentence sequence $\{s_1, s_2, ..., s_n\}$, we define a coherence measure based on the average of sentence co-occurrence scores, namely, $sco(r) = \frac{1}{n-1} \sum_{i=1}^{n-1} sco(s_i, s_{i+1})$, where $sco(s_i, s_{i+1})$ can be instantiated by Eqn. 8 or Eqn. 9. We denote the corresponding coherence measures as sco^{as} and sco^{bs} respectively.

Recall that synthetic reviews are essentially generated by combining sentences from multiple reviews. We expect abnormal word co-occurrence phenomena in synthetic reviews, i.e., two words observed in a consecutive sentence pair of a synthetic review hardly co-occur in that of truthful reviews. Therefore, the measure SCO tends to give a lower value on synthetic reviews than on truthful reviews.

4.3 Pairwise Sentence Similarity

Pairwise sentence similarity takes a different approach to measure the transition of sentences. It focuses on the similar words or topics that are shared by two consecutive sentences. There are multiple ways to measure similarity of two sentences. Here we briefly introduce three mechanisms:

Word overlap: One baseline similarity measure computes the proportion of overlapped words in two sentences [12]: $wolap\_sim(s_1, s_2) = \frac{2|s_1 \cap s_2|}{|s_1|+|s_2|}$.

WordNet-based word similarity: WordNet [18] groups English words into sets of cognitive synonyms (named *synset*). Each word can belong to multiple synsets. A WordNet-based similarity measure can derive a similarity score at the semantic level. For any two sentences s_1 and s_2, the WordNet-based similarity measure is defined as $\frac{1}{|s_1||s_2|} \sum_{w_i \in s_1, w_j \in s_2} \max_{c_i, c_j} sim(c_i, c_j)$, where c_i (c_j) is one of the synsets that word w_i (w_j) can belong to. $sim(c_i, c_j)$ is the path similarity between two synsets [18]. We denote this measure as $wonet\_sim(s_1, s_2)$

Latent semantic similarity: Latent Semantic Indexing (LSI)[4] employs Singular Value Decomposition (SVD) to uncover the latent semantic patterns in the usage of words. Based on our training review dataset, we first formulate a matrix D of size $N \times M$, where N is the number of words and M is the total number of sentences. $D_{i,j}$ is the term frequency-inverse document frequency (tf-idf) of word w_i in sentence s_j. Through SVD, we obtain $D = USV^T$, where $U^{N \times K}$ reflects the main K patterns of word co-occurrences. For a test sentence s, we compute its projection to the latent LSI space determined by the K principal components:

$$L_s = U^T s \quad (10)$$

Given a sentence pair (s_1, s_2), their latent semantic similarity is determined by: $lsi\_sim(s_1, s_2) = \cos(L_{s_1}, L_{s_2})$, where L_{s_1} and L_{s_2} are obtained according to Eqn. 10.

Given any of the above similarity measures, for a review r, the coherence score is defined as the average pairwise-sentence similarity over all the consecutive sentence pairs, i.e., $\frac{1}{n-1} \sum_{i=1}^{n-1} sim(s_i, s_{i+1})$.

4.4 Multiple Sentence Coherence

Following the discussion in Section 3, running length can be directly instantiated using any pairwise sentence similarity measure. For the semantic dispersion measure, we can instantiate it based on Latent Semantic Indexing: For each sentence in a review, we first obtain a vector representation v_i according to Eqn. 10. Then Eqn. 1 is directly applicable to obtain SD. This LSI-instantiated measure is denoted as SD^{lsi}.

Our experiments show that multiple versions of running length using pairwise similarity functions discussed in the previous section, do not work very well. It is partially because the synthesis process uses similar sentences as replacement so that deceptive reviews share similar topic variations with truthful reviews. As a future study, it would be interesting to redefine the concept of running length to accommodate measures such as sentence transition and word co-occurrence.

5. EXPERIMENTS

In this section, we first evaluate the effectiveness of our proposed methodology on detecting synthesized reviews. Then we extend the application of our coherence measures to ranking reviews based on their authenticity.

5.1 Experimental Setup

In practice, reviews are not naturally labeled. To test various deception detectors, we need to create experimental datasets with ground truth. We collected $12,500$ reviews of hotels located in New York City. These reviews have five scores and contain more than 150 characters. Based on this collection, 10 datasets are created: For each dataset, we first randomly sample 500 reviews from the collection and treat them as base reviews. The remaining $12,000$ are put into the review pool. For each of the 500 base reviews, we synthesize one review following the pipeline shown in Figure 2. A single dataset is composed of 500 synthetic reviews and 500 base reviews, and totally 10 such datasets are created. Multiple ways to construct the experimental datasets might exist; we select a most straightforward, but nontrivial one as we will show later.

A subset of our instantiated measures, such as PTP, SCO, and LSI-related measures, require a dataset to learn parameters in their model. We form an additional dataset with $12,000$ new reviews. In order to avoid any detection bias, we allow no overlap among this review set and the previously constructed datasets.

Both the review synthesis process and detection algorithms were implemented in Python. All the experiments were performed on a 2.67GHZ, 12GB, Intel PC running Fedora 13.

5.2 Classification

For each review, we first extract all the coherence features discussed in Section 4, and transform a review to a feature vector representation. These review vectors are input to a classifier to tell if one review is truthful or synthetic. In our experiments, three classifiers, SVM with a linear kernel (linearSVM) and a polynomial kernel (polynomialSVM), and Naive Bayes (NB) classifier are employed. We use a public data mining software, Weka [9], to run all the classification tasks. All the parameters in the classifiers are set at default except that we choose a quadratic kernel in polynomialSVM.

On each of the 10 datasets, we conduct a 5-fold cross-validation procedure. Following conventions in supervised learning literature, we use three measures to evaluate the detection performance on each dataset: (1) misclassification error rate (ER), i.e., the percentage of misclassified reviews over all reviews; (2) true positive rate (TPR); (3) false positive rate (FPR). In computation of TPR and FPR, we treat truthful reviews as positive instances. We finally show the average performance over the 10 datasets.

5.3 Evaluation of Coherence Measures

Unless otherwise stated, we pick $K = 300$ in LSI-related measures. Detection results based on our coherence measures are summarized in Table 2. *Pair.*, *Multi.*, and *Comb.* respectively denote pairwise sentence, multiple sentence coherence measures, and combinations of them. The three classifiers generally achieve consistent performance. Among the four variants of PTP, the pivot edge measure ptp^e and mixture measure ptp^m achieve much better performance than the other two. For SCO, the average score variant sco^{as} performs comparably to ptp^e and ptp^m. Potential reasons for PTP and SCO to work well have been discussed in Section 4.

For all the variants of SIM, we only show the range of their error rate and omit TPR and FPR due to space constraint. In contrast with PTP and SCO, SIM measures do not perform as well as one might expect. This could be due to the fact that our synthesis algorithm uses similar sentences as replacement, making similarity based measures ineffective. We next provide a formal analysis. Given a review r with a sentence sequence $\{s_1, s_2, \ldots, s_n\}$, each sentence can be viewed as a point in a high dimensional space. Here we use a 2-d space plot (Figure 6) to illustrate these points. The solid line shows the transition of sentences (denoted as black points) in review r; the dash line shows the transition of sentences (denoted as white points) in the synthesized review r'. For each point s_i, the synthesis algorithm will find a close point s'_i in the review pool. In the following theorem, we show that if the distance between s_i and s'_i is small enough, the average distance between two consecutive sentences in r' would not be significantly different from the one in r.

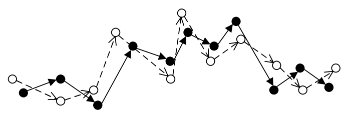

Figure 6: Sentence Similarity

Theorem 1: Let d be a distance function between two sentences, satisfying the triangle inequality. Given two reviews $r = (s_1, s_2, \ldots, s_n)$ and $r' = (s'_1, s'_2, \ldots, s'_n)$, for $\delta > 0$, if $\forall$ i, $d(s_i, s'_i) \leq \delta$, then $|m - m'| \leq 2\delta$ and $|\sigma^2 - \sigma'^2| \leq 8\sigma\delta + 16\delta^2$, where m (m') and $\sigma^2(\sigma'^2)$ are the average and variance of the pairwise sentence distance in r (r') respectively.

Proof: Let $l_i = d(s_i, s_{i+1})$ and $l'_i = d(s'_i, s'_{i+1})$. According to the triangle inequality, we have:

$$|l_i - l'_i| \leq d(s_i, s'_i) + d(s_{i+1}, s'_{i+1}) \leq 2\delta$$

	Instantiated measures		linearSVM			polynomialSVM			NB		
			ER (%)	TPR	FPR	ER (%)	TPR	FPR	ER (%)	TPR	FPR
Pair.	PTP	ptp^n	52.2	0.53	0.57	53.0	0.60	0.66	49.4	0.77	0.75
		ptp^{be}	41.8	0.83	0.66	49.0	0.83	0.81	49.4	0.80	0.78
		ptp^e	41.2	0.71	0.54	38.2	0.80	0.57	40.3	0.84	0.65
		ptp^m	38.6	0.65	0.42	41.2	0.60	0.43	36.7	0.66	0.39
	SCO	sco^{as}	39.0	0.68	0.46	39.0	0.74	0.52	38.7	0.78	0.55
		sco^{bs}	49.4	0.54	0.52	49.6	0.58	0.57	48.4	0.70	0.66
	SIM	all variants	ER (%): 48.9-49.8			ER (%): 48.7-50.3			ER (%): 49.4-50.1		
	PTP+SCO		33.3	0.71	0.5	37.4	0.78	0.53	35.4	0.85	0.56
Multi.	SD	SD^{lsi}	41.5	0.68	0.51	39.3	0.74	0.52	32.7	0.89	0.55
	RL	all variants	ER (%): 49.0-50.1			ER (%): 50.1-50.2			ER (%): 49.2-49.3		
Comb.	PTP+SD		22.9	0.82	0.28	23.0	0.82	0.28	27.9	0.89	0.45
	SCO+SD		34.9	0.70	0.40	32.0	0.72	0.36	26.5	0.84	0.38
	PTP+SCO+SD		**21.6**	0.82	0.25	**22.0**	0.80	0.25	**24.5**	0.86	0.35
Our competitors			Error rate (%)			TPR			FPR		
Ott *et al.* [19]			40.5			0.68			0.32		
Liu *et al.* [15]			*34.5*			0.66			0.35		
Harris *et al.* [10]			43.3			0.57			0.45		
Human			48.0			0.60			0.56		

Table 2: Performance of Coherence Measures.

	wolap_sim		*wonet_sim*		*lsi_sim*		SD^{lsi}	
	m	σ	m	σ	m	σ	m	σ
Truthful	0.05	0.04	0.84	0.42	0.10	0.08	2.07	0.37
Synthetic	0.05	0.04	0.89	0.48	0.10	0.08	2.37	0.76

Table 3: The Mean (m) and Standard Deviation (σ) of Coherence Measures.

then the difference between the average pairwise distance in r and r' satisfies

$$|m - m'| = \tfrac{1}{n-1}|\textstyle\sum_{i=1}^{n-1}(l_i - l'_i)| \leq 2\delta$$

Similarly,

$$\begin{aligned} |\sigma^2 - \sigma'^2| &= \tfrac{1}{n-1}|\sum_{i=1}^{n-1}(l_i - m)^2 - (l'_i - m')^2| \\ &\leq \tfrac{1}{n-1}\sum_{i=1}^{n-1}(|l_i - l'_i| + |m - m'|)\cdot|l_i - m + l'_i - m'| \\ &\leq \tfrac{4\delta}{n-1}\sum_{i=1}^{n-1}(2|l_i - m| + |l'_i - l_i| + |m - m'|) \\ &\leq 8\delta\sum_{i=1}^{n-1}\tfrac{|l_i - m|}{n-1} + 16\delta^2 \\ &\leq 8\delta\sqrt{\sum_{i=1}^{n-1}\tfrac{|l_i - m|^2}{n-1}} + 16\delta^2 \\ &\leq 8\sigma\delta + 16\delta^2 \end{aligned}$$

□

This proof gives us the intuition that, measured by any distance metric, if s_i and s'_i are close to each other, those coherence features measured by the same distance function, might not work well in synthetic review detection, because they tend to be similar on synthetic reviews and truthful reviews. The above analysis can partially explain the poor performance of SIM. As a validation, it is clear that on synthetic reviews the mean and standard deviation of SIM measures in Table 3, are quite close to those on truthful reviews.

For multiple sentence measures, semantic dispersion turns out to be effective. We verify the difference between the distribution of SD^{lsi} on truthful and synthetic reviews in Table 3. Running length (RL) does not work well, partially due to the fact that their instantiated measures are related to similarity measures used to synthesize reviews.

	Error rate (%)		
	linearSVM	polynomialSVM	NB
SD^{ℓ_2}	41.5	39.3	32.7
$SD^{\cos}$	50.6	50.9	50.6

Table 4: Semantic Dispersion Using SD^{ℓ_2} and $SD^{\cos}$.

Enlightened by Theorem 1, we attribute the success of SD to that the Euclidean distance measure (i.e., ℓ_2 norm) is significantly different from the cosine similarity measures in our review synthesis model. To empirically show this, we conduct one control test: Replace the Euclidean distance in SD with cosine similarity, i.e., replace $\|v_i - centroid\|$ with $\cos(v_i, centroid)$ in Eqn.1 (denoted as $SD^{\cos}$), and compare $SD^{\cos}$ with the original SD (denoted as SD^{ℓ_2}). The results are shown in Table 4, from which we observe that replacing the Euclidean distance with cosine similarity severely undermines the detection accuracy.

To further enhance the detection performance, we select the most effective variant from each measure category to combine with each other. As shown in Table 2, the combination of PTP and SCO only gives around 3%-5% improvement compared with either of them. However, combining pairwise sentence measures with the multiple sentence measure SD significantly decreases the error rate to around 22%. This result is reasonable, in that PTP and SCO measure the pairwise (local) coherence from a similar perspective while SD tries to capture the coherence among multiple sentences

Methods	Error rate (%)	TPR	FPR
PTP+SCO+SD	**26.7**	0.77	0.31
Ott *et al.* [19]	39.6	0.62	0.41
Liu *et al.* [15]	41.3	0.56	0.38
Harris *et al.* [10]	43.4	0.52	0.39

Table 5: Adaptability Study of Different Methods.

(globally) from a different perspective: These two kinds of measures complement each other to enhance the detection performance. Compared with the state-of-the-art spam detectors, PTP+SCO+SD significantly reduces the error rate by roughly 13%, improves the true positive rate from 0.68 to 0.82 while decreasing the false positive rate from 0.32 to 0.25. We further compare the receiver operating characteristic, or ROC curve, of PTP+SCO+SD (under linearSVM) with that of our competing methods in Figure 7. The area under the curve obtained by our method is much larger than that by other methods, indicating that our method possesses better predictive power of truthful and synthetic reviews.

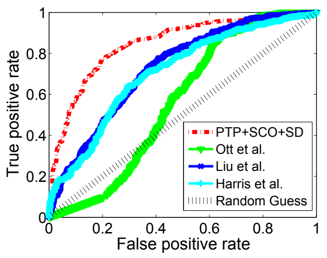

Figure 7: ROC Curves of Different Methods.

When performing spam detection in a review forum, it is possible that no training datasets from that forum are available. In such cases, one desideratum is that a deception detector can still perform well although it is trained on a not-so-relevant dataset. Such deception detectors are regarded as adaptable. To test the adaptability of different methods, we employ one of the previous 10 datasets for training and an additional dataset for testing. This additional dataset was constructed in the same manner as before, except using reviews from Washington D.C. instead of New York City. We repeat training-testing for 10 times by changing the training dataset each time. The average performance for each method is shown in Table 5. For PTP+SCO+SD, linearSVM is used for classification since it performs best in previous experiments. In this adaptability study, our method still achieves the best error rate, and significantly improves the true positive rate while depressing the false positive rate.

5.4 Ranking Reviews w.r.t Their Authenticity

Apart from binary classification of reviews, in real-life applications, it could also be useful to present users a list of reviews in descending order of authenticity. We refer this problem as review ranking. Here we show how to apply our coherence measures to rank reviews, and evaluate their effectiveness based on the quality of the sorted review list. Let r be the set of features for a review with label y (truthful or synthetic), we quantify the authenticity of each review as the posterior probability $P(y = truthful|r)$, which is directly obtained from the output of Naive Bayes classifier.

All the reviews are further sorted in descending order of $P(y = truthful|r)$. With a different feature set, one review can be determined authentic to a different degree, therefore resulting in a different rank in the list. In our experiments, review ranking is evaluated by the ratio of truthful reviews in the top K review list, denoted as precision@K. We first calculate precision@K on each dataset and show the averaged result over the 10 datasets. To compare our measures with others, we also implement the features proposed in [19, 15, 10] as the feature set and obtain $P(y = truthful|r)$ based on the output of Naive Bayes classifier. The performance of these features is shown in Table 6. When increasing the length (K) of the review list, our proposed measure PTP+SCO+SD consistently achieves over 0.9 precision, indicating that our proposed features also work well for the review ranking task.

6. RELATED WORK

Our work is mainly related to previous studies in three categories: (1) Review spam detection; (2) Text reuse detection; (3) Text quality study.

Review spam detection. Review spam (or opinion spam) tries to mislead readers by composing untruthful views, which has been studied recently in [11, 13, 19, 6]. In [11, 13], researchers focus on detecting disruptive review spam such as reviews with irrelevant texts by utilizing information such as statistical features from review texts, behaviors of review spammers, and relationships among reviewers. Feng *et al.* [6] detects those reviews as opinion spam which distort the underlying background distribution of opinions. Recent work by Ott *et al.* [19] studies deceptive reviews written by Amazon Mechanical Turkers based on n-gram and psychological deception features. To enhance the deception detection performance, Harris *et al.* [10] further combines the algorithm by Ott *et al.* [19] with easily-obtained statistical information from the review text.

In this work, we emphasize the potential threat of automatically synthesized reviews. These reviews are extremely hard to detect because sentences in each review were written by people who had true experiences. Neither coarse-grained statistical features such as number of words in a review nor abnormal writing patterns of liars can be employed to tell the difference between truthful and synthetic reviews.

Text reuse detection. Text reuse refers to repetitively using parts of the texts in previously created documents. Extensive research studies on text reuse detection have been conducted in the web search context. Examples include the detection of duplicate or near-duplicate documents [20, 3], and phrase-level duplication [7]. More recently, researchers in [1] identify reused and modified local texts on the web such as sentences or passages instead of whole documents.

Our review synthesization bears similarity with text reuse, in that each review is generated by combining sentences from other existing reviews. However, as we discussed in Section 2.4, to pass sentence-level text reuse detectors, one could further use paraphrasing techniques [17, 8]. Besides, local text content is not a unique fingerprint of one specific review. It might cause high false positive rate if one utilizes the presence of reused texts as an indicator of a deceptive review.

Text quality study. In our review synthesis model, fake

	PTP	SCO	SD	PTP+SCO+SD	Ott *et al.* [19]	Liu *et al.* [15]	Harris *et al.* [10]
precision@20	0.87	0.69	0.61	**0.98**	0.62	0.72	0.82
precision@50	0.87	0.73	0.60	**0.97**	0.47	0.79	0.84
precision@100	0.85	0.72	0.63	**0.96**	0.47	0.80	0.82
precision@200	0.80	0.69	0.64	**0.93**	0.52	0.78	0.77

Table 6: Performance of Review Ranking: Precision of Top-K Reviews

reviews, formed by sentence replacements, might result in low-quality writing in terms of sentence connection or coherence. Research studies such as [15, 16] focus on classifying reviews as helpful (considered as high quality) or unhelpful (considered as low quality) based on human judgements. Features employed by those approaches do not particularly account for the coherence of a review's text, which, however, is the key factor to detect synthetic reviews in our setting. Authors of [12, 14] discuss automatic coherence evaluation of a general text using various sentence similarity measures or discourse relations, some of which have been employed to instantiate our framework. Our new coherence measures turn out to outperform both the low quality detection features and those sentence similarity measures.

7. CONCLUSION

In this paper, we bring into attention a simple yet powerful review synthesis technique, which could be employed by evil attackers for large scale spamming. Furthermore, we propose a general framework to defend the review communities against such automatically synthesized reviews. Compared with existing deception detectors, the instantiated framework with our new coherence measures can significantly improve the detection performance by roughly 13%. While our method achieves the initial success, it is still an open research problem to further improve the detection accuracy. One meaningful extension is to study the prevalence of synthesized reviews in real review environment.

8. ACKNOWLEDGEMENTS

This research was sponsored in part by the U.S. National Science Foundation under grant IIS-0917228. The views and conclusions contained herein are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notice herein.

9. REFERENCES

[1] M. Bendersky and W. Croft. Finding text reuse on the web. In *WSDM*, pages 262–271, 2009.

[2] D. Blei, A. Ng, and M. Jordan. Latent dirichlet allocation. *JMLR*, 3:993–1022, 2003.

[3] A. Broder. Identifying and filtering near-duplicate documents. In *Combinatorial Pattern Matching*, pages 1–10. Springer, 2000.

[4] S. Deerwester, S. Dumais, G. Furnas, T. Landauer, and R. Harshman. Indexing by latent semantic analysis. *Journal of the American society for information science*, 41(6):391–407, 1990.

[5] R. Fano. Transmission of information: A statistical theory of communications. *American Journal of Physics*, 29:793–794, 1961.

[6] S. Feng, L. Xing, A. Gogar, and Y. Choi. Distributional footprints of deceptive product reviews. In *AAAI on Weblogs and Social Media*, 2012.

[7] D. Fetterly, M. Manasse, and M. Najork. Detecting phrase-level duplication on the world wide web. In *SIGIR*, pages 170–177, 2005.

[8] J. Ganitkevitch, C. Callison-Burch, C. Napoles, and B. Van Durme. Learning sentential paraphrases from bilingual parallel corpora for text-to-text generation. In *EMNLP*, pages 1168–1179, 2011.

[9] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. Witten. The weka data mining software: an update. *SIGKDD Explorations*, 11(1):10–18, 2009.

[10] C. Harris. Detecting deceptive opinion spam using human computation. In *Workshops at AAAI on AI*, 2012.

[11] N. Jindal and B. Liu. Opinion spam and analysis. In *WSDM*, pages 219–230, 2008.

[12] M. Lapata and R. Barzilay. Automatic evaluation of text coherence: Models and representations. In *IJCAI*, volume 19, page 1085, 2005.

[13] E.-P. Lim, V.-A. Nguyen, N. Jindal, B. Liu, and H. W. Lauw. Detecting product review spammers using rating behaviors. In *CIKM*, pages 939–948, 2010.

[14] Z. Lin, H. Ng, and M. Kan. Automatically evaluating text coherence using discourse relations. In *ACL*, pages 997–1006, 2011.

[15] J. Liu, Y. Cao, C. Lin, Y. Huang, and M. Zhou. Low-quality product review detection in opinion summarization. In *EMNLP-CoNLL*, pages 334–342, 2007.

[16] Y. Liu, X. Huang, A. An, and X. Yu. Modeling and predicting the helpfulness of online reviews. In *ICDM*, pages 443–452. IEEE, 2008.

[17] N. Madnani and B. Dorr. Generating phrasal and sentential paraphrases: A survey of data-driven methods. *Computational Linguistics*, 36(3):341–387, 2010.

[18] G. Miller and C. Fellbaum. Wordnet: An electronic lexical database, 1998.

[19] M. Ott, Y. Choi, C. Cardie, and J. T. Hancock. Finding deceptive opinion spam by any stretch of the imagination. In *ACL-HLT*, pages 309–319, 2011.

[20] B. Stein, M. Koppel, and E. Stamatatos. Plagiarism analysis, authorship identification, and near-duplicate detection pan'07. In *ACM SIGIR Forum*, volume 41, pages 68–71. ACM, 2007.

[21] J. Stribling, M. Krohn, and D. Aguayo. Scigen-an automatic cs paper generator, 2005. *http://pdos.csail.mit.edu/scigen.*