Collusive Opinion Fraud Detection in Online Reviews: A Probabilistic Modeling Approach

CHANG XU and JIE ZHANG, Nanyang Technological University

We address the collusive opinion fraud problem in online review portals, where groups of people work together to deliver deceptive reviews for manipulating the reputations of targeted items. Such collusive fraud is considered much harder to defend against, since the participants (or colluders) can evade detection by shaping their behaviors collectively so as not to appear suspicious. To alleviate this problem, countermeasures have been proposed that leverage the collective behaviors of colluders. The motivation stems from the observation that colluders typically act in a very synchronized way, as they are instructed by the same campaigns with common items to target and schedules to follow. However, the collective behaviors examined in existing solutions focus mostly on the external appearance of fraud campaigns, such as the campaign size and the size of the targeted item set. These signals may become ineffective once colluders have changed their behaviors collectively. Moreover, the detection algorithms used in existing approaches are designed to only make collusion inference on the input data; predictive models that can be deployed for detecting emerging fraud cannot be learned from the data. In this article, to complement existing studies on collusive opinion fraud characterization and detection, we explore more subtle behavioral trails in collusive fraud practice. In particular, a suite of homogeneity-based measures are proposed to capture the interrelationships among colluders within campaigns. Moreover, a novel statistical model is proposed to further characterize, recognize, and predict collusive fraud in online reviews. The proposed model is fully unsupervised and highly flexible to incorporate effective measures available for better modeling and prediction. Through experiments on two real-world datasets, we show that our method outperforms the state of the art in both characterization and detection abilities.

CCS Concepts: • **Applied computing** → **Secure online transactions**; • **Information systems** → *Reputation systems;*

Additional Key Words and Phrases: Collusive fraud, opinion manipulation, fraud detection, probabilistic modeling, prediction and inference

ACM Reference Format:
Chang Xu and Jie Zhang. 2017. Collusive opinion fraud detection in online reviews: A probabilistic modeling approach. ACM Trans. Web 11, 4, Article 25 (July 2017), 28 pages.
DOI: http://dx.doi.org/10.1145/3098859

1. INTRODUCTION

As online reviews have become increasingly influential in helping online shoppers make purchase decisions, opinion fraud [9] has emerged as a major threat to this process. This blackhat practice is intended to affect people's buying decisions by creating misleading reviews about particular items (e.g., products and restaurants). By committing opinion

This work is supported by the MOE AcRF Tier 2 Grant M4020110.020. This article is a revised and extended version of an article that appeared in the *Proceedings of the 2015 IEEE International Conference on Data Mining*, retrieved from http://ieeexplore.ieee.org/abstract/document/7373434/.
Authors' addresses: C. Xu (corresponding author) and J. Zhang, School of Computer Science and Engineering, Nanyang Technological University, 50 Nanyang Avenue, Block N4, #02C-100, 639798, Singapore; emails: xuch0007@e.ntu.edu.sg, ZHANGJ@ntu.edu.sg.

DOI: http://dx.doi.org/10.1145/3098859

fraud, malicious business owners can often achieve sales increase by posting false-positive reviews for themselves or leaving false-negative reviews for their rivals. It has been estimated that about 16% of Yelp reviews written for the restaurants in the metropolitan Boston area are fake [14]. To make the situation even worse, opinion fraud practitioners today have evolved in specialization; they are found to collaborate and form coordinated campaigns [17, 27] such that more manpower and trickier tactics can be put into use to achieve more covert and cost-effective fraud practices.

There have been prior attempts at tackling such *collusive fraud* based on data mining and machine-learning techniques. Supervised approaches are proposed for detecting review fraud campaigns by training models on annotated fake review data [21, 27]. Although shown to possess high detection accuracy, these methods rely heavily on real fake reviews for model learning, which are difficult to obtain in practice due to the lack of ground truth in real-world scenarios [25]. To overcome this problem, Mukherjee et al. [17] pioneer the exploration of a variety of *collective behavior* patterns of users and propose an unsupervised framework to score users in an iterative manner. A ranking list of users can then be formed according to the scores users obtain; the higher the ranks, the more suspicious the corresponding users. Ye and Akoglu [29] study the network-based characteristics of colluders under the same campaigns and cluster them based on their commonly targeted items. Nevertheless, a major problem is that these approaches cannot learn from existing data for making predictions about emerging collusion events. In fact, predictive models are crucial to fraud detection tasks, as they are deployable in real-time scenarios where emerging fraud practices can be detected and removed in a timely manner to minimize the damages. Moreover, being a deterministic model by nature, the existing approaches could be sensitive to the variability in the feature computation that can occur for different reasons, such as particular parameter settings or noisy input data.

In this work, we are motivated to fill those gaps by modeling collusive fraud in online reviews. In particular, we identify the problem of detecting collusive opinion fraud from a stochastic perspective and seek a statistical solution that can (1) infer collusion in existing unlabeled data as unsupervised models, (2) learn to make collusion predictions as supervised models, and (3) handle uncertainty in the measurements of engineered features. In the light of such objectives, we propose the Latent Collusion Model (LCM), a novel statistical model that fulfills all the above goals. The key perspective of LCM is based on the appreciation of revealed characteristics that can differentiate colluders[1] from non-colluders in specific feature space (e.g., References [17, 27]). Colluders have been found to exhibit *unique collective behavior* patterns that result from their collaborations. It is then possible for LCM to build models for both tasks of collusion inference and prediction by taking a hybrid of generative and discriminative approaches. Specifically, for the collusion inference task, the goal is to infer the class labels of users (i.e., colluders or non-colluders) in the provided data, and the generative approach can model the process that generates the observed collective behaviors from the user class labels. This enables LCM to formulate collusion inference as a clustering problem where collusions can be inferred by clustering distinctive collective behaviors of colluders and non-colluders in the feature space. On the other hand, for the collusion prediction task, one can learn a collusion predictive model by taking the discriminative approach, which models the dependence of the user class labels on the observed collective behaviors directly. This renders LCM to learn a predictive distribution over user class labels, which will be used to identify potential collusions in future reviewing events.

[1]We use the term "colluder" to refer to spammers who are involved in collusive opinion fraud.

Moreover, two variants of LCM are explored and studied. The first is a chaining-based model that performs the collusion inference and prediction tasks in a sequential manner. This model decouples the operations of the two tasks and enables separate training of specialized models that are suitable for the respective tasks. The second is a unified model that combines the learning and inference of the generative and discriminative approaches to achieve the optimal detection accuracy via balancing between the two views. The experiments show that both the variants have their own merits in meeting different detection demands and can complement each other in terms of collusion inference and prediction performance. Finally, by modeling all the involved entities, that is, user class labels and feature dimensions, as random variables and their relationships as probabilistic distributions in LCM, the uncertainty lurk in the input data and the model building process can be captured and handled in a coherent manner.

To complement existing collusion-oriented features and expand the feature space for collusive fraud characterization, a suite of *homogeneity-based* collusive behavior measures (h-CBMs) are also proposed to distinguish colluders from non-colluders. Differing from existing features and signals used for modeling colluders' collective behaviors, h-CBMs focus on the internal properties of fraud campaigns and capture the intrinsic connections between colluders by measuring various behavioral similarities. Through h-CBMs, we find that to complete the tasks assigned by a campaign, colluders tend to take very similar actions such as targeting almost the same items, providing highly consistent review ratings, and posting reviews at proximate time points. To avoid detection, even though colluders can restrict the sizes of launched campaigns or overwhelm the targets by review-flooding, the actions they take to finish the common tasks are inevitably homogeneous, which renders them detectable.

It is also worth noting that as no restriction is imposed on the feature dimension involved in the model building, LCM can benefit from other effective collusion features apart from the proposed h-CBMs. This can be seen as another advantage of LCM over other unsupervised methods (e.g., References [2, 17]), in which the detection algorithms are hard coded with specific model assumptions or algorithmic constraints and cannot incorporate additional feature dimensions.

Summary of Contributions. We address the collusive opinion fraud detection problem in online reviews from a probabilistic modeling perspective, originally introduced in Reference [26]. In this extended work, our contributions are augmented to a novel variant of the proposed statistical model and several homogeneity-based collusive behavior measures from a larger spectrum of evaluations using newer datasets. In summary, this work offers the following contributions:

- A novel statistical model, named LCM is proposed to defend against collusive opinion fraud that is (a) unsupervised (no annotation of collusive fraud is needed), (b) predictive (knowledge about collusive behavior patterns can be learned for collusion forecasts), and (c) flexible (can benefit from potential collusion features for better model building).
- Two variants of the proposed statistical model are explored and studied. The chaining-based model decouples the collusion inference and prediction procedures, which enables separate training of specialized models that are suitable for respective tasks. The unified model combines the learning and inference of the generative and discriminative approaches in a coherent and principled manner to achieve optimal detection performance via balancing between the generative and discriminative views. We show that both variants of the proposed statistical model have their own merits in meeting different detection scenarios and can complement each other in terms of collusion inference and prediction performance.

- Principled optimization algorithms are developed to perform posterior inference and parameter estimation for LCM, which are shown to converge fast in the experiments.
- A suite of novel collusive behavior measures are proposed to capture the intrinsic homophily in collusive opinion fraud from various reviewing behavioral dimensions. We show that with these features, we can effectively detect not only blatant fraud campaigns but also stealth ones with small sizes and a restrained targeting spectrum.
- The effectiveness of the proposed collusion measures is extensively evaluated, and detailed comparison analysis between the proposed LCM and competitive alternatives on two real-world datasets is offered.

2. PRELIMINARIES

In this section, we introduce the notions and definitions that are related to the problem of collusive opinion fraud detection. Differing from general opinion fraud, collusive opinion fraud is one particular type where multiple spammers are asked to write fake reviews for the same set of items. Such collective practices are often formalized and organized as spam campaigns.

Assume that in a typical review site there is a set of users $\mathcal{V} = \{v\}$ writing reviews for a set of items $\mathcal{I} = \{i\}$. For each user v, let $\mathcal{I}_v$ be the set of items reviewed. For each review written by user v for item i, let $r_{v,i}$ and $t_{v,i}$ be the rating and timestamp associated with this review.[2]

Definition 2.1 (*General Opinion Fraud and Spammer*). In general, opinion fraud involves a **spammer** being asked to write a fake review to a designated item, with the goal of either promoting or demoting the overall sentiment towards the item.

Definition 2.2 (*Collusive Opinion Fraud, Spam Campaign, and Colluder*). In collusive opinion fraud, a **spam campaign** involves the collective efforts of a group of spammers targeting the same set of items. To highlight the difference from the general spammers, the members of spam campaigns are called **colluders**.

Definition 2.3 (*Candidate Colluder Group*). To capture the collective behaviors of colluders, we borrow the concept of **candidate colluder group** from Reference [17]. A **candidate colluder group** refers to a group of users who have reviewed multiple items together.[3]

Note that a candidate colluder group is not necessarily a true colluder group, since normal users can also review the same items due to similar preferences. However, it is a necessary condition for a user to become a colluder. *Given the notion of candidate colluder group, our goal of collusive opinion fraud detection then reduces to differentiating colluder groups from non-colluder groups*.

Definition 2.4 (*Colluder Group*). A **colluder group** consists of colluders involved in the same spam campaigns, while a **non-colluder group** consists of either normal users or the mix of normal users and colluders.[4] Note that a spam campaign can correspond to one or more colluder groups.

[2]In this work, we do not utilize the text of review for model building, mainly because (1) not every real-world review site would provide such information, and (2) review text is vulnerable to manipulation [23]. From this perspective, our approach is a complement to existing methods that favor textual features of fraudulent reviews (e.g., References [19, 20]).

[3]In Reference [17], the equivalent term used is "candidate spammer group." The authors used frequent itemset mining (FIM) to generate groups consisting of at least two users who have reviewed at least three items together.

[4]Our assumption of treating the mixed groups that contain both colluders and normal users as negatives (i.e., non-colluder groups) is based on the consideration that the colluders and normal users in the mixed

Given the above definition, we use g to denote a candidate colluder group and $\mathcal{M}_g$ for its members. Similarly to $\mathcal{I}_v$ representing the set of items reviewed by a user v, we use $\mathcal{I}_g$ to represent the set of items reviewed by a group g.

3. CHARACTERING COLLUSIVE OPINION FRAUD

Effective capture of collective behaviors of colluders plays an important role in collusive opinion fraud detection. In this section, we propose our approach to characterizing collusive fraud in online reviews.

As discussed, colluders from the same campaigns can exhibit unique collective behavior patterns. However, existing features may not be able to tackle colluders of all kinds. Some of these features are designed mainly for finding large campaigns that involve a non-trivial number of colluders and targets. For example, *Group Size* and *Group Support Count* proposed in Reference [17] assume groups having larger size and attacking a greater number of targets are more suspicious. However, these features are likely to miss stealthier attacks where large campaigns split into smaller ones for different subsets of targets. Also, there are features regarding colluders as anomaly and measuring how much their behaviors would deviate from those of general users. For example, *Group Deviation* proposed in Reference [17] evaluates the difference between a group and the remaining population on a target in terms of review rating. The larger the deviation between the average rating given by a group and the remaining population, the more likely this group is a colluder group. However, due to their advantage in manpower, colluders could easily turn themselves into the majority by review-flooding the targets, so the deviations can be eliminated.

To cope with the above issues, we develop a suite of h-CBMs to inspect the homogeneity in the collective behaviors of colluders. In other words, our focus is on the similar behaviors of colluders exhibited during their working for the same campaigns. Specifically, the proposed h-CBMs are imposed on the candidate colluder groups, and the characterization is based on four behavioral dimensions that release strong signals related to collusive opinion fraud. They are related to the items reviewed by a group (Target-based h-CBMs), the ratings given by a group (Rating-based h-CBMs), the timestamps of the reviews given by a group (Temporal-based h-CBMs), and the member activity degree of a group (Activity-based h-CBMs).

3.1. Target-based h-CBMs

First and foremost, a significant characteristic of collusive opinion fraud would be related to the common items reviewed by colluders. Since no spam campaigns will launch without specifying any targeted items, and colluders are required to review these items to get paid, the connections among colluders can be best revealed by inspecting the common items they focus on. Intuitively, colluders working for the same campaigns would be assigned to the same targets, which would inevitably leave common segments in colluders' reviewing histories. One thus can trace the collusion frequency of a group by counting the number of common items reviewed by its members. The higher the count, the more likely the members are to commit collusive fraud together. Here, we use Jaccard Similarity for such a **target consistency (TC)** measurement. Specifically, we inspect the Jaccard Similarity of the set of items reviewed by the members of group g as follows:

$$h_g^{TC} = \frac{\#\{\cap_{v \in \mathcal{M}_g} \mathcal{I}_v\}}{\#\{\cup_{v \in \mathcal{M}_g} \mathcal{I}_v\}}, \tag{1}$$

groups are irrelevant with their own underlying motivations behind. There should be no difference between groups that contain only normal users and the mixed ones.

where #{·} evaluates the cardinality of a set, the numerator is the number of common items reviewed by all the members $\mathcal{M}_g$, and the denominator serves as a normalizer by computing the number of items reviewed by either of them. This similarity is essentially evaluating the concentration of a group of users on their common targets.

3.2. Rating-Based h-CBMs

The review rating an item receives conveys the overall opinion from the general public and thus becomes the target of manipulation. Since the ultimate objective of fraud campaigns is to promote or demote the targeted items, colluders are required to give highly positive or negative ratings to a particular target together to quickly lift up or bring down its average rating. It thus should be suspicious if a group of users often provide consistently high/low ratings to their commonly reviewed items. To capture this, we inspect the **rating consistency (RC)** of a user group g by computing the variance of the ratings given by its members $\mathcal{M}_g$ to the set of commonly reviewed items $\mathcal{I}_g$; the lower the variance, the more consistent the provided ratings are, and the more suspicious the group should be. The formula for computing RC is

$$h_g^{RC} = \max_{i \in \mathcal{I}_g}(\text{var}(\{r_{v,i} | v \in \mathcal{M}_g\})). \tag{2}$$

We can see that Equation (2) first evaluates the variance of ratings given by a group g to each of the common items in $\mathcal{I}_g$. It then chooses the maximum variance due to the consideration that in general colluders' ratings would be consistent on most of their common items, such that the maximum variance would be still quite low for most of the time. In contrast, the rating variances of a non-colluder group should be randomly distributed, rendering the maximum value more likely to be relatively larger.

3.3. Temporal-Based h-CBMs

Timing is vital to fraud campaigns, since efficiency usually brings more profits. Moreover, to quickly achieve the desired manipulation effect, a fraud campaign will typically set up a schedule and the campaign members are required to finish their tasks in time. This, ironically, can be used to our advantage as the imposed time constraint on campaign activities would restrict the actions of colluders within a bounded time interval. Measures based on a time window can therefore be used to capture such synchronization of colluders' behaviors. We then employ the **temporal synchronization (TS)** measure to inspect the review posting timing of a group of users on each of the targeted items,

$$h_g^{TS} = \max_{i \in \mathcal{I}_g}(\text{var}(\{t_{v,i} | v \in \mathcal{M}_g\})). \tag{3}$$

Similarly to the rating consistency discussed before, Equation (3) first computes the variance of the review posting timestamp of group g on each of its targeted items in $\mathcal{I}_g$. It then chooses the maximum variance due to the same reason as the case in the rating consistency measure. A lower TS value will indicate that a group is more suspicious for its synchronized reviewing behavior in time.

In addition to the absolute review posting timestamps, the relative orders of review postings can also be used to reveal the orchestration of a fraud campaign. The utilization of review order comes in two flavors. First, it can consolidate the capture of review synchronization, because the reviews posted at proximate times are more likely to be placed near to each other on the review pages. Second, as most of the compromised items would be the unpopular ones with few reviews received [14] (otherwise, there is no reason for reputation manipulation), even though the campaign period can be longer in some cases, resulting in larger time gaps between consecutive colluder reviews, the

relative orders of the colluder reviews would be still quite close to each other, as it would be less likely that many truthful reviews are injected between the consecutive colluder reviews on unpopular items. In such cases, the consistency of review orders becomes more important to indicate the occurrence of collusive fraud. We thus introduce the **review order synchronization (ROS)** measure to evaluate the synchronization of the orders of colluder reviews as follows:

$$h_g^{ROS} = \max_{i \in \mathcal{I}_g}(\text{var}(\{o_{v,i} | v \in \mathcal{M}_g\})). \tag{4}$$

We can see that Equation (4) takes a similar form with Equation (3) by computing the maximum over the variances of review orders of a group on common item set $\mathcal{I}_g$.

Another time-related collusion signal is the time when a colluder posts his or her first review. It is reported that general spammers tend to use accounts that are auto-generated or bought in batches for their first campaign practice [15]. Thus it is possible to spot another type of synchronicity by comparing the time when colluders use their accounts for the first time. Specifically, we propose the **first-review synchronization (FS)** measure as the variance of timestamps of the first reviews posted by the members of group g as follows:

$$h_g^{FS} = \text{var}(\{\min(\{t_{v,i} | i \in \mathcal{I}_v\}) | v \in \mathcal{M}_g\}), \tag{5}$$

where the function $\min(\{t_{v,i} | i \in \mathcal{I}_v\})$ retrieves the time when a group member posts his or her first review.

3.4. Activity-Based h-CBMs

The posting activeness of colluders can also reveal the dynamics of a particular spam campaign due to theer being a requirement to follow the same campaign schedule and also share a part of the overall posting workload, which will make their activity patterns very similar. Specifically, in terms of the campaign schedule factor, we propose the **activity consistency (AC)** measure to compute the closeness between the most active moments of the group members. This is done by evaluating the variance of the most active moments of the group members. The most active moment of a user is defined as the date on which he or she posts most of his or her reviews. The formulation is as follows:

$$h_g^{AC} = \text{var}(\{t_v^{max} | v \in \mathcal{M}_g\}), \tag{6}$$

where $t_v^{max} = \arg\max_{t \in T_v}(\#\{t_{v,i} = t | i \in \mathcal{I}_v\})$ retrieves the most active moment of user v.

To account for the workload sharing factor, one way to measure the workload would be to count the total number of reviews posted by a user. To share the overall workload of a campaign, the workload assigned to each colluder may be similar. In this case, we measure the **workload similarity (WS)** as the variance of the numbers of items reviewed by the members in group g as

$$h_g^{WS} = \text{var}(\{|\mathcal{I}_v| | v \in \mathcal{M}_g\}), \tag{7}$$

where $\mathcal{I}_v$ is the set of reviews posted by user v.

Finally, we may also observe colluders' correlation in terms of their account usage pattern. As colluders may use particular accounts for particular campaigns (or multiple campaigns), the accounts used for the same campaign(s) may experience similar lifetimes. We can then compare the lifetimes of the members' accounts in a group. In particular, the lifetimes of the accounts used for the same campaigns are expected to be of similar length. We employ the **account lifetime similarity (ALS)** measure, which evaluates the variance of the account lifetimes of group members. The account

lifetime of a user is defined as the time span over his or her first and last reviews,

$$h_g^{ALS} = \text{var}(\{\max(T_v) - \min(T_v) | v \in \mathcal{M}_g\}), \tag{8}$$

where $T_v = \{t_{v,i} | i \in \mathcal{I}_v\}$ is the set of timestamps of user v's all postings.

Numerically, the value of each h-CBM is standardized using 0-1 scaling and thus in the real interval [0,1]. The variance-based h-CBMs (i.e., all except TC) are converted to similarity-based measures by using the formula

$$s_{\text{new}} = \frac{2}{1 + s_{\text{old}}} - 1, \tag{9}$$

where s_{old} and s_{new} are the values of a particular h-CBM before and after the conversion. After the conversion, for all h-CBMs, a value closer to 1 will suggest that the evaluated group is more likely to be a colluder group.

It is worth noting that a merit of h-CBMs is that they are parameter-free. This omits the need for parameter estimation based on held-out labeled data, given the fact that high-quality annotation for opinion fraud detection is hard to obtain in practice.

4. PROBLEM FORMULATION

In this section, we formulate our target problem of collusive fraud detection in online reviews. In a review site, we are given a set of N candidate colluder groups $\mathcal{G} = \{g_1, \ldots, g_N\}$, in which each group g_n is associated with an M-dimensional h-CBM[5] vector $\boldsymbol{\phi}_n = \{\phi_n^{(1)}, \ldots, \phi_n^{(M)}\}$, $\phi_n^{(m)} \in [0, 1]$. Then the set of all h-CBM vectors $\boldsymbol{\Phi} = \{\boldsymbol{\phi}_1, \ldots, \boldsymbol{\phi}_N\}$ constitutes our observed data about all the groups $\mathcal{G}$. The class label of each group g_n is denoted by a binary variable $z_n \in \{0, 1\}$, specifying whether it is a colluder group or not. As our context is unsupervised, all the class labels $\boldsymbol{Z} = \{z_1, \ldots, z_N\}$ of $\mathcal{G}$ are unknown.

Now we define the problem of detecting collusive opinion fraud from a *probabilistic modeling* perspective as follows:

PROBLEM 1. ***Given*** *a set of candidate colluder groups $\mathcal{G}$ with their observed h-CBM vectors $\boldsymbol{\Phi}$, the problem of detecting collusive fraud in online reviews involves two subtasks: (1)* ***infer*** *the posterior distribution of the class labels $\boldsymbol{Z}$ of $\mathcal{G}$, $p(\boldsymbol{Z}|\boldsymbol{\Phi})$, and (2)* ***predict*** *the class label $\hat{z}$ for an emerging group $\hat{g}$ based on the predictive distribution $p(\hat{z}|\hat{\boldsymbol{\phi}}, \boldsymbol{\Phi}, \boldsymbol{Z})$, where $\hat{\boldsymbol{\phi}}$ is the h-CBM vector of $\hat{g}$.*

5. THE PROPOSED STATISTICAL MODEL

In this section, we propose a novel statistical model, named LCM, to solve the problem of collusive opinion fraud detection from a probabilistic perspective.

The key idea of LCM is to treat the unknown class labels of candidate colluder groups $\boldsymbol{Z}$ as latent and consider the *reciprocal* relationship between the class labels $\boldsymbol{Z}$ and the observed h-CBM vectors $\boldsymbol{\Phi}$ from two probabilistic modeling views: generative and discriminative. In general, the generative view considers the class labels as the *hidden factor* that causes the distinctive h-CBM distributions of colluder groups and normal groups, which allows us to perform the collusion inference task by taking a clustering-based generative approach that defines a generative process for generating group h-CBM distributions from the class labels. On the other hand, the discriminative view regards the class labels as the *hidden consequence* of the observed h-CBM distributions, which enables the learning of a predictive distribution for the collusion prediction tasks.

[5]Note that other collusion-oriented features can also be used in LCM. The term "h-CBM" here is for representational purpose only.

To further take advantage of the two view modeling, two variants of LCM are proposed. The first variant is a chaining-based model that performs the two subtasks of collusion inference and prediction by chaining the generative and discriminative views in sequence. The second one is a unified model that combines both views in an integrated framework. We will show that both the variants have their own merits and can complement each other in terms of collusion inference and prediction performance. For both variants, principled optimization algorithms are implemented to conduct model learning and inference.

5.1. Two Views of Collusive Opinion Fraud Modeling

As discussed, the goal of modeling collusive opinion fraud is to solve Problem 1 involving the two subtasks of collusion inference and prediction. Specifically, for the collusion inference task, the objective is to compute the posterior distribution $p(\boldsymbol{Z}|\boldsymbol{\Phi})$, while for the collusion prediction task, we need to derive the predictive distribution $p(\hat{z}|\hat{\boldsymbol{\phi}}, \boldsymbol{\Phi}, \boldsymbol{Z})$. It can be seen that in both cases the unknown quantity $\boldsymbol{Z}$ plays an important role and serves as the connection between the two subtasks. As such, we can treat the class labels $\boldsymbol{Z}$ as latent in LCM and further consider the *reciprocal* relationship between the unknown class labels $\boldsymbol{Z}$ and the observed h-CBM vectors $\boldsymbol{\Phi}$ from two probabilistic modeling views: generative and discriminative.

5.1.1. The Generative View: Occurred Collusion Inference. The generative view of LCM considers the class labels as the *hidden factor* that causes the h-CBM vectors. As mentioned, colluders often possess unique collective behavior patterns in feature space. Once we know the class label of a group, we can somehow generate its h-CBM vector based on the difference between the collective behavior patterns (distributions) of colluder and non-colluder groups, which essentially yields two distinctive clusters. Then we can take a clustering-based generative approach [3] to infer the posterior distribution $p(\boldsymbol{Z}|\boldsymbol{\Phi})$ to solve the inference task of Problem 1.

Specifically, the generative approach for this clustering postulates a generative process describing how the M dimensional h-CBM vectors of a mixture of N colluder and non-colluder groups can be generated by LCM:

(1) For each group g_n, where $n \in \{1, \ldots, N\}$, generate its class label z_n based on a cluster membership distribution p_z: $z_n \sim p_z$, where $z \in \{0, 1\}$ and $p_1 + p_0 = 1$;
(2) For each h-CBM dimension $\phi_n^{(m)}$ of group g_n, where $m \in \{1, \ldots, M\}$, generate the h-CBM value from the cluster-conditional distribution parametrized as $p(\phi_n^{(m)}|\boldsymbol{\lambda}_{z_n}^{(m)})$: $\phi_n^{(m)} \sim p(\phi_n^{(m)}|\boldsymbol{\lambda}_{z_n}^{(m)})$.

The parameter for the cluster-conditional distribution $\boldsymbol{\lambda}_{z_n}^{(m)} = \{\boldsymbol{\lambda}_0^{(m)}, \boldsymbol{\lambda}_1^{(m)}\}$ essentially captures the latent collective behavior patterns of colluders ($\boldsymbol{\lambda}_1^{(m)}$ with $z_n = 1$) and non-colluders ($\boldsymbol{\lambda}_0^{(m)}$ with $z_n = 0$) on the mth h-CBM dimension. As each h-CBM dimension $\phi_n^{(m)}$ takes values in [0, 1], we substantialize the cluster-conditional distribution $p(\phi_n^{(m)}|\boldsymbol{\lambda}_{z_n}^{(m)})$ as a standard Beta distribution $\mathbf{Beta}(\phi_n^{(m)}|\alpha_{z_n}^{(m)}, \beta_{z_n}^{(m)})$ with parameters $\alpha_{z_n}^{(m)}$ and $\beta_{z_n}^{(m)}$, such that $\boldsymbol{\lambda}_{z_n}^{(m)} = [\alpha_{z_n}^{(m)}, \beta_{z_n}^{(m)}]^T$.

Given this generative process, the likelihood of generating the observed data $\boldsymbol{\Phi}$ based on the model parameter $\boldsymbol{\lambda} = \{\boldsymbol{\lambda}_0^{(m)}\}_{m=1}^M \cup \{\boldsymbol{\lambda}_1^{(m)}\}_{m=1}^M$ can be written as

$$L(\boldsymbol{\lambda}|\boldsymbol{\Phi}, \boldsymbol{Z}) = \prod_{n=1}^{N} \left[p_1 \cdot \prod_{m=1}^{M} \mathbf{Beta}\left(\phi_n^{(m)}|\alpha_1^{(m)}, \beta_1^{(m)}\right) \right]^{z_n} \left[p_0 \cdot \prod_{m=1}^{M} \mathbf{Beta}\left(\phi_n^{(m)}|\alpha_0^{(m)}, \beta_0^{(m)}\right) \right]^{1-z_n}. \tag{10}$$

Then we are able to compute the posterior distribution $p(\boldsymbol{Z}|\boldsymbol{\Phi}, \boldsymbol{\lambda})$, which is the goal of collusion inference, by maximizing the likelihood in Equation (10) with respect to the model parameter $\boldsymbol{\lambda}$ using an expectation-maximization (EM) algorithm [5]. More details of the optimization algorithms will be presented later in Sections 5.2 and 5.3.

5.1.2. The Discriminative View: Emerging Collusion Prediction. To model the relationship between the colluder group class labels and h-CBMs, the discriminative view, on the other hand, regards the class labels as the *hidden consequence* of the observed h-CBM vectors. This view is considered more natural, since the very goal of h-CBMs is to discriminate between the collective behavioral patterns of colluders and non-colluders; given the h-CBM vector of a particular group, we can tell its class label by referring to the semantics of each h-CBM dimension.

To solve the collusion prediction task of Problem 1, that is, to compute the predictive distribution $p(\hat{z}|\hat{\boldsymbol{\phi}}, \boldsymbol{\Phi}, \boldsymbol{Z})$, we take a discriminative approach [3], where $p(z_n|\boldsymbol{\phi}_n)$ is directly defined. Here we use a logistic function to model this distribution:

$$p(z_n = 1|\boldsymbol{\phi}_n, \mathbf{w}) = \sigma(\mathbf{w}^T \boldsymbol{\phi}_n), \tag{11}$$

where $\sigma(x) = \frac{1}{1+\exp(-x)}$ is the logistic function; $\mathbf{w} = \{\mathrm{w}_0, \ldots, \mathrm{w}_M\}$ is an $(M+1)$-dimensional weight vector to be estimated; and w_0 is the weight for an additional dummy h-CBM $\boldsymbol{\phi}_n^{(0)} = 1$, which is added for notational compactness. In general, other suitable models for binary responses could also apply, such as the probit regression model [4]. Then, the predictions for a future group $\hat{g}$ with $\{\hat{\boldsymbol{\phi}}, \hat{z}\}$ can be made by marginalizing the predictive distribution with respect to the model parameter w:

$$p(\hat{z} = 1|\hat{\boldsymbol{\phi}}, \boldsymbol{\Phi}, \boldsymbol{Z}) = \int p(\hat{z} = 1|\hat{\boldsymbol{\phi}}, \mathbf{w}) p(\mathbf{w}|\boldsymbol{\Phi}, \boldsymbol{Z}) \mathrm{d}\mathbf{w}. \tag{12}$$

However, this marginalization requires the knowledge of all class labels $\boldsymbol{Z}$, and as our context is unsupervised, we are not provided with such information. Later we will show how to obtain this information by taking advantage of the generative view discussed before. Now, given the generative and discriminative views of the collusive fraud modeling, in the subsequent sections, we will introduce two variants of LCM that build probabilistic models based on these two views.

5.2. The Chaining-Based Model

A straightforward and intuitive way of solving Problem 1 from a probabilistic perspective would be to tackle the two subtasks, that is, collusion inference and prediction, in sequel. The idea is that as the collusion prediction task in Problem 1 acquires class labels for predictive distribution estimation, we can thus perform the collusion inference task first, which can produce the two clusters of colluder and non-colluder groups through the clustering-based generative approach described in Section 5.1.1. Then, by regarding the inferred cluster labels as the *pseudo* group class labels, we are then able to estimate the predictive distribution in the collusion prediction task based on the observed group h-CBM vectors and the inferred pseudo class labels. As the operations of the two tasks of collusion inference and prediction are decoupled during the process, the advantage of such a chaining-based modeling is the inherent flexibility that specialised models suitable for the respective tasks can be optimized and applied separately. The full chaining procedure for the collusion inference and prediction is listed as follows:

(1) Prepare the input data by computing the h-CBM vectors $\boldsymbol{\Phi} = \{\boldsymbol{\phi}_1, \ldots, \boldsymbol{\phi}_N\}$ for a set of N candidate groups $\mathcal{G} = \{g_1, \ldots, g_N\}$ whose class labels $\boldsymbol{Z} = \{z_1, \ldots, z_N\}$ need to be inferred;

(2) Perform **collusion inference** task: to infer the posterior distribution of the class labels $\boldsymbol{Z}$: $q(\boldsymbol{Z}|\boldsymbol{\Phi})$;
(3) Obtain the "pseudo" class labels $\boldsymbol{Z}$ of candidate groups according to the decision rule: for each $g_n \in \mathcal{G}$, we have:

$$z_n = 1 \quad \text{if} \quad q(z_n = 1|\boldsymbol{\phi}_n) > q(z_n = 0|\boldsymbol{\phi}_n) \quad \text{else} \quad 0;$$

(4) Perform **collusion prediction** task: to estimate the parameter $\mathbf{w}$ of the predictive distribution $p(\hat{z}|\hat{\boldsymbol{\phi}}, \boldsymbol{\Phi}, \boldsymbol{Z})$ based on $\boldsymbol{Z}$ obtained in step 3;
(5) Output the inferred group class labels $\boldsymbol{Z}$ for $\mathcal{G}$, and the parameter of the predictive distribution $\mathbf{w}$.

Now we provide details of computing the posterior distribution $q(\boldsymbol{Z}|\boldsymbol{\Phi})$ in step 2 for collusion inference and estimate the parameter $\mathbf{w}$ of the predictive distribution in step 4 for collusion prediction, respectively.

First, we compute the posterior distribution $q(\boldsymbol{Z}|\boldsymbol{\Phi})$ according to the generative view of LCM discussed in Section 5.1.1. In particular, the likelihood of generating the observed data $\boldsymbol{\Phi}$ based on the model parameter $\boldsymbol{\lambda} = \{\boldsymbol{\lambda}_0^{(m)}\}_{m=1}^M \cup \{\boldsymbol{\lambda}_1^{(m)}\}_{m=1}^M$, $\boldsymbol{\lambda}_{(\cdot)}^{(m)} = [\alpha_{(\cdot)}^{(m)}, \beta_{(\cdot)}^{(m)}]^T$ is given in Equation (10). As $\boldsymbol{Z}$ is latent in the model, we develop an EM algorithm to perform parameter estimation and posterior inference. Specifically, based on Equation (10), the complete data log-likelihood of this generative model can be written as

$$\begin{aligned}\mathcal{L}_{\text{collusion\_inference}}(\boldsymbol{\lambda}|\boldsymbol{\Phi}, \mathbf{Z}) = &\sum_{n=1}^{N} z_n \left[\log p_1 + \log \sum_{m=1}^{M} \mathbf{Beta}\big(\phi_n^{(m)}|\alpha_1^{(m)}, \beta_1^{(m)}\big) \right] \\ &+ \sum_{n=1}^{N} (1 - z_n) \left[\log p_0 + \log \sum_{m=1}^{M} \mathbf{Beta}\big(\phi_n^{(m)}|\alpha_0^{(m)}, \beta_0^{(m)}\big) \right].\end{aligned} \tag{13}$$

Thus, in the E-step, we evaluate the posterior distribution of z_n given the current parameter values as

$$\begin{aligned}q(z_n = 1) = \mathbb{E}[z_n] &= \frac{p_1 \prod_{m=1}^{M} \mathbf{Beta}(\phi_n^{(m)}|\alpha_1^{(m)}, \beta_1^{(m)})}{\sum_{k\in\{0,1\}} p_k \prod_{m=1}^{M} \mathbf{Beta}(\phi_n^{(m)}|\alpha_k^{(m)}, \beta_k^{(m)})} \\ q(z_n = 0) &= 1 - q(z_n = 1).\end{aligned} \tag{14}$$

In the M-step, we estimate the parameters of the model based on $\{q(z_n)\}_{n=1}^N$ by maximizing the log-likelihood. In particular, the update equation for the cluster membership distribution $p_z, z \in \{0, 1\}$ is

$$p_1 = \frac{1}{N} \sum_{n=1}^{N} q(z_n = 1), \qquad p_0 = \frac{1}{N} \sum_{n=1}^{N} q(z_n = 0). \tag{15}$$

For the parameter $\boldsymbol{\lambda} = \{\alpha_k^{(m)}, \beta_k^{(m)}\}_{m=1}^M$, $k \in \{0, 1\}$, the optimal value $\{\alpha_k^{(m)*}, \beta_k^{(m)*}\}$ for each parameter pair are coupled and need to be solved simultaneously. We resort to the Newton-Raphson method where each Beta parameter pair $\lambda_k^{(m)} = \{\alpha_k^{(m)}, \beta_k^{(m)}\}$ can be optimized iteratively:

$$\lambda_k^{(m)\text{new}} = \lambda_k^{(m)\text{old}} - \mathbf{H}_{\lambda_k}^{-1} \mathbf{g}_{\lambda_k} \tag{16}$$

with gradients $\mathbf{g}_{\lambda_k} = [g^{\alpha}_{\lambda_k}, g^{\beta}_{\lambda_k}]$:

$$\begin{aligned} g^{\alpha}_{\lambda_k} &= \sum_n q(z_n = k)\big\{\log \phi_n^{(m)} - \big[\psi\big(\alpha_k^{(m)}\big) - \psi\big(\alpha_k^{(m)} + \beta_k^{(m)}\big)\big]\big\} \\ g^{\beta}_{\lambda_k} &= \sum_n q(z_n = k)\big\{\log\big(1 - \phi_n^{(m)}\big) - \psi\big(\beta_k^{(m)}\big) - \psi\big(\alpha_k^{(m)} + \beta_k^{(m)}\big)\big]\big\} \end{aligned} \tag{17}$$

and Hessian $\mathbf{H}_{\lambda_k}$:

$$\mathbf{H}_{\lambda_k} = \sum_n q(z_n = k) \cdot \begin{bmatrix} \psi'\big(\alpha_k^{(m)} + \beta_k^{(m)}\big) - \psi'\big(\alpha_k^{(m)}\big) & \psi'\big(\alpha_k^{(m)} + \beta_k^{(m)}\big) \\ \psi'\big(\alpha_k^{(m)} + \beta_k^{(m)}\big) & \psi'\big(\alpha_k^{(m)} + \beta_k^{(m)}\big) - \psi'\big(\beta_k^{(m)}\big) \end{bmatrix}, \tag{18}$$

where $\psi(\cdot)$ is the *digamma* function and $\psi'(\cdot)$ the *trigamma* function. After the EM algorithm converges, we obtain the posterior distribution of the class labels and further the pseudo class label $\hat{\boldsymbol{Z}} = \{\hat{z_n}\}$ according to step 3.

Next, given the pseudo group class labels $\hat{\boldsymbol{Z}}$, we estimate the parameter $\mathbf{w}$ of the predictive distribution for collusion prediction. In particular, the log-likelihood of the discriminative view is given by

$$\mathcal{L}_{\text{collusion\_prediction}} = \sum_{n=1}^{N} \hat{z_n} \log \sigma(\mathbf{w}^T \boldsymbol{\phi}_n) + (1 - \hat{z_n}) \log(1 - \sigma(\mathbf{w}^T \boldsymbol{\phi}_n)). \tag{19}$$

Due to the nonlinearity of logistic function, the optimal solution cannot be found analytically. Again, we appeal to the Newton-Raphson method where the optimal $\mathbf{w}^*$ can be estimated iteratively:

$$\mathbf{w}^{\text{new}} = \mathbf{w}^{\text{old}} - \mathbf{H}_{\mathbf{w}}^{-1} g_{\mathbf{w}}, \tag{20}$$

with gradient and Hessian given by

$$\begin{aligned} g_{\mathbf{w}} &= \sum_{n=1}^{N} \big(\sigma\big(\mathbf{w}^{\text{old}^T} \boldsymbol{\phi}_n\big) - \hat{z_n}\big)\boldsymbol{\phi}_n \\ \mathbf{H}_{\mathbf{w}} &= \sum_{n=1}^{N} \sigma\big(\mathbf{w}^{\text{old}^T} \boldsymbol{\phi}_n\big)\big(1 - \sigma\big(\mathbf{w}^{\text{old}^T} \boldsymbol{\phi}_n\big)\big)\boldsymbol{\phi}_n \boldsymbol{\phi}_n^T. \end{aligned} \tag{21}$$

5.3. The Unified Model

In this subsection, we introduce the second model variant that utilizes the two-view perspective for collusive fraud detection. In fact, a more coherent and principled approach for solving Problem 1 would be to combine the generative and discriminative views into a unified framework, such that the class labels and the predictive distribution can be obtained at the same time.

Although possessing the ability to take advantage of unlabeled data for collusion inference, the generative view of LCM suffers from the common problem of a generative probabilistic model that the performance would deteriorate if the generative model were not a perfect match for the true data generation process in reality [11]. When it comes to the chaining-based model described above, the inaccuracy of the generative view modeling would subsequently affect the performance of the collusion prediction procedure that takes the inferred cluster labels from the generative view as input. On the other hand, by directly modeling the dependence of the class labels on the data, that is, $p(\boldsymbol{Z}|\boldsymbol{\Phi})$, the discriminative view of LCM would achieve better results when a modeling mismatch exists. However, due to the fact that the discriminative view ignores the structure $p(\boldsymbol{\Phi}|\boldsymbol{Z})$, it cannot effectively learn the colluder behavioral pattern

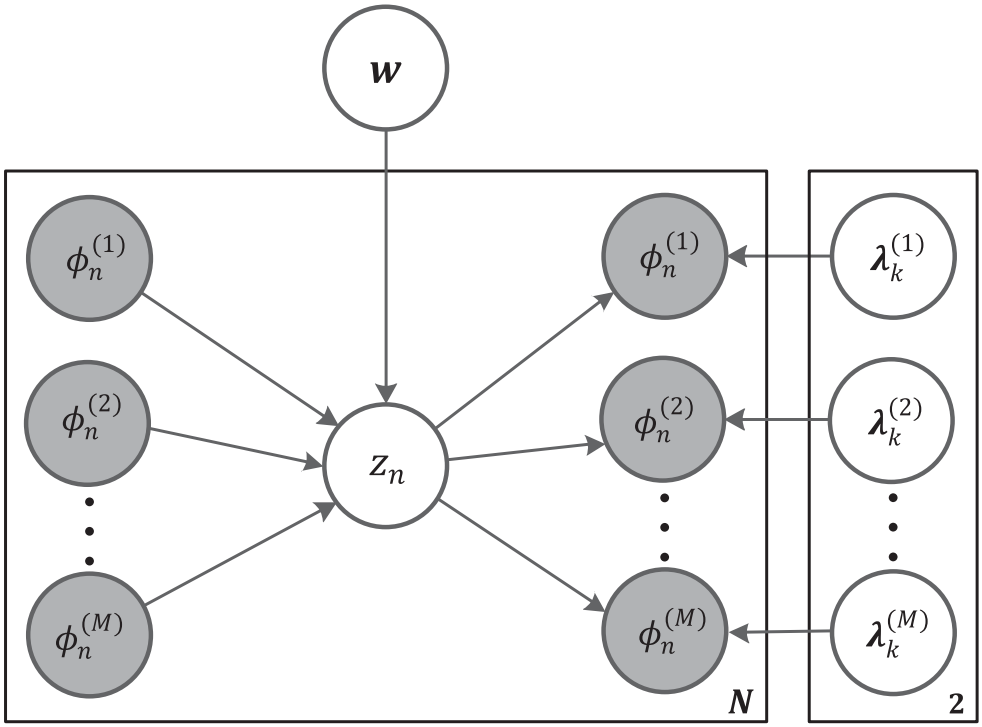


Fig. 1. Graphical model representation of unified LCM. Light circles denote the latent random variables $\{z_n\}_{n=1}^N$ and model parameters $\{\mathbf{w}, \boldsymbol{\lambda}\}$, and dark circles denote the observed random variables $\{\boldsymbol{\phi}_n\}_{n=1}^N$. The left plate indexed by N denotes the groups in $\mathcal{U}$, and the right plate indexed by "2" denotes the h-CBM distributions for colluder ($k = 1$) and non-colluder ($k = 0$) groups in $\mathcal{U}$.

from the data, which can be captured by the distribution of the Beta parameter $\boldsymbol{\lambda}$ in Equation (13) introduced in the generative view of LCM.

Therefore, to benefit from the merits of both generative and discriminative views, we would expect that an optimal trade-off can be found by combining the two views in a concurrent and principled way. In particular, we design a hybrid of generative and discriminative models according to the principle of blending generative and discriminative models suggested in Reference [12]. The graphical model representation of the unified LCM is depicted in Figure 1. The idea is to model the joint probability of all the h-CBM vectors $\boldsymbol{\Phi}$ and the class labels $\mathbf{Z}$, given the model parameters $\{\mathbf{w}, \boldsymbol{\lambda}\}$, written as follows:

$$
\begin{aligned}
p(\boldsymbol{\Phi}, \mathbf{Z}|\mathbf{w}, \boldsymbol{\lambda}) &= \prod_{n=1}^{N} p(z_n|\boldsymbol{\phi}_n, \mathbf{w}) \prod_{m=1}^{M} p\big(\phi_n^{(m)}|\boldsymbol{\lambda}_{z_n}^{(m)}\big) \\
&= \prod_{n=1}^{N} \left[\frac{1}{1+\exp(-\mathbf{w}^T\boldsymbol{\phi}_n)}\right]^{z_n} \left[\frac{\exp(-\mathbf{w}^T\boldsymbol{\phi}_n)}{1+\exp(-\mathbf{w}^T\boldsymbol{\phi}_n)}\right]^{1-z_n} \\
&\quad \cdot \left(\prod_{m=1}^{M} \left[\frac{\Gamma(\alpha_1^{(m)}+\beta_1^{(m)})}{\Gamma(\alpha_1^{(m)})\Gamma(\beta_1^{(m)})} \big(\phi_n^{(m)}\big)^{\alpha_1^{(m)}-1} \big(1-\phi_n^{(m)}\big)^{\beta_1^{(m)}-1}\right]^{z_n}\right. \\
&\quad \left. \cdot \left[\frac{\Gamma(\alpha_0^{(m)}+\beta_0^{(m)})}{\Gamma(\alpha_0^{(m)})\Gamma(\beta_0^{(m)})} \big(\phi_n^{(m)}\big)^{\alpha_0^{(m)}-1} \big(1-\phi_n^{(m)}\big)^{\beta_0^{(m)}-1}\right]^{1-z_n}\right).
\end{aligned}
\tag{22}
$$

We can see that compared to the model specification in the generative view (Equation (10)), the cluster membership distribution p_z is replaced by the discriminative classification distribution $p(z_n|\boldsymbol{\phi}_n, \mathbf{w})$ parametrized by the predictive distribution parameter $\mathbf{w}$. This replacement thus connects the two views of modeling in a way that the estimation of the discriminative model parameter $\mathbf{w}$ and generative model parameter $\boldsymbol{\lambda}$ will affect each other through the inference about the latent class label variable $\boldsymbol{Z}$ during model optimization, which we will show in the subsequent section.

5.3.1. Learning and Inference. The model optimization involves the computation of the posterior distribution of latent variables and the estimation of the model parameters. In the unified LCM, the latent variables $\mathbf{Z} = \{z_n\}_{n=1}^N$ account for the occurrence of collusion in the input data. The model parameters are $\{\mathbf{w}, \boldsymbol{\lambda}\}$, where $\mathbf{w}$ enables the learning of a collusion predictor, and $\boldsymbol{\lambda} = \{\boldsymbol{\lambda}_0^{(m)}\}_{m=1}^M \cup \{\boldsymbol{\lambda}_1^{(m)}\}_{m=1}^M$ captures the collective behavior patterns of colluders and non-colluders. Again, as the unified LCM is a latent variable model, we develop an EM algorithm to perform model learning and inference. Specifically, we first derive the complete data log-likelihood of the model by taking the logarithm of Equation (22):

$$\begin{aligned}\mathcal{L}_{\text{unified}}(\mathbf{w}, \boldsymbol{\lambda}|\boldsymbol{\Phi}, \mathbf{Z}) &= \log p(\boldsymbol{\Phi}, \mathbf{Z}|\mathbf{w}, \boldsymbol{\lambda}) \\ &= \sum_{n=1}^N z_n \log \sigma\left(\mathbf{w}^T \boldsymbol{\phi}_n\right) + (1 - z_n) \log\left(1 - \sigma\left(\mathbf{w}^T \boldsymbol{\phi}_n\right)\right) \\ &\quad + \sum_{n=1}^N z_n \log \sum_{m=1}^M \mathbf{Beta}\left(\phi_n^{(m)} | \alpha_1^{(m)}, \beta_1^{(m)}\right) \\ &\quad + \sum_{n=1}^N (1 - z_n) \log \sum_{m=1}^M \mathbf{Beta}\left(\phi_n^{(m)} | \alpha_0^{(m)}, \beta_0^{(m)}\right).\end{aligned} \tag{23}$$

E-step: We derive the posterior of latent variable $\mathbf{Z}$ based on Bayes' theorem and Equation (22) as follows:

$$\begin{aligned}q(\mathbf{Z}|\boldsymbol{\Phi}, \mathbf{w}, \boldsymbol{\lambda}) &\propto p(\boldsymbol{\Phi}, \mathbf{Z}|\mathbf{w}, \boldsymbol{\lambda}) \\ &= \prod_{n=1}^N \left[\sigma\left(\mathbf{w}^T \boldsymbol{\phi}_n\right) \prod_{m=1}^M \mathbf{Beta}\left(\phi_n^{(m)} | \alpha_1^{(m)}, \beta_1^{(m)}\right)\right]^{z_n} \\ &\quad \cdot \left[\left(1 - \sigma\left(\mathbf{w}^T \boldsymbol{\phi}_n\right)\right) \prod_{m=1}^M \mathbf{Beta}\left(\phi_n^{(m)} | \alpha_0^{(m)}, \beta_0^{(m)}\right)\right]^{1 - z_n}.\end{aligned} \tag{24}$$

For simplicity, it is assumed that each group is independent with each other and so is its label. Then, by factorizing Equation (24), we obtain the posterior of each z_n as

$$q(z_n = 1|\boldsymbol{\phi}_n, \mathbf{w}, \boldsymbol{\lambda}) = \sigma\left(\mathbf{w}^T \boldsymbol{\phi}_n + \triangle_{\boldsymbol{\lambda}}\right) \tag{25}$$

with $\triangle_{\boldsymbol{\lambda}}$ defined as

$$\begin{aligned}\triangle_{\boldsymbol{\lambda}} &= \sum_{m=1}^M \log \Gamma\left(\alpha_1^{(m)} + \beta_1^{(m)}\right) + \log \Gamma\left(\alpha_0^{(m)}\right) + \log \Gamma\left(\beta_0^{(m)}\right) \\ &\quad - \log \Gamma\left(\alpha_0^{(m)} + \beta_0^{(m)}\right) - \log \Gamma\left(\alpha_1^{(m)}\right) - \log \Gamma\left(\beta_1^{(m)}\right) \\ &\quad + \left(\alpha_1^{(m)} - \alpha_0^{(m)}\right) \log \phi_n^{(m)} + \left(\beta_1^{(m)} - \beta_0^{(m)}\right) \log\left(1 - \phi_n^{(m)}\right).\end{aligned}$$

We can see that in Equation (25) the inference about the class labels $\boldsymbol{Z} = \{z_n\}$ requires the knowledge about the generative parameter $\boldsymbol{\lambda}$ and the discriminative parameter $\mathbf{w}$.
M-step: We find the estimate for model parameters $\{\mathbf{w}, \boldsymbol{\lambda}\}$ to maximize the expected value of the complete data log-likelihood $\mathbb{E}_{\boldsymbol{Z}}[\log p(\boldsymbol{\Phi}, \mathbf{Z}|\mathbf{w}, \boldsymbol{\lambda})]$.

For estimating the discriminative parameter $\mathbf{w}$, similarly to the case in the chaining-based model, we appeal to the Newton-Raphson method where the optimal $\mathbf{w}^*$ can be estimated iteratively:

$$\mathbf{w}^{\text{new}} = \mathbf{w}^{\text{old}} - \mathbf{H}_{\mathbf{w}}^{-1} g_{\mathbf{w}}, \tag{26}$$

with gradient $g_{\mathbf{w}} = \boldsymbol{\Phi}^T(\boldsymbol{q}^{\bar{z}} - \boldsymbol{p}^{\bar{z}})$ and Hessian $\mathbf{H}_{\mathbf{w}} = \boldsymbol{\Phi}^T \boldsymbol{Q} \boldsymbol{\Phi}$, where $\boldsymbol{q}^{\bar{z}}$ is a column vector $\{\mathbb{E}_q[z_n]\}_{n=1}^N$, $\boldsymbol{p}^{\bar{z}}$ a column vector $\{\mathbb{E}_p[z_n]\}_{n=1}^N$, and $\boldsymbol{Q}$ an $N \times N$ diagonal matrix with elements $Q_{nn} = \mathbb{E}_q[z_n](1 - \mathbb{E}_q[z_n])$. Note that, differing from the estimation of $\mathbf{w}$ in the chaining-based model in Equation (20), the computation of gradient and Hessian here requires the expectation of the class labels $\{z_n\}$ with respect to their posterior distributions $\{q(z_n)\}$, while in Equation (20) the hard assignments of the class labels are used instead.

For estimating generative parameter $\boldsymbol{\lambda} = \{\alpha_k^{(m)}, \beta_k^{(m)}\}_{m=1}^M$, with $k \in \{0, 1\}$, the optimal value $\{\alpha_k^{(m)*}, \beta_k^{(m)*}\}$ for each parameter pair are coupled and need to be solved simultaneously. Similarly, the Newton-Raphson method is used where each Beta parameter pair $\lambda_k^{(m)} = \{\alpha_k^{(m)}, \beta_k^{(m)}\}$ can be optimized iteratively:

$$\lambda_k^{(m)\text{new}} = \lambda_k^{(m)\text{old}} - \mathbf{H}_{\lambda_k}^{-1} \mathbf{g}_{\lambda_k}, \tag{27}$$

where gradients $\mathbf{g}_{\lambda_k} = [g_{\lambda_k}^{\alpha}, g_{\lambda_k}^{\beta}]$,

$$\begin{aligned} g_{\lambda_k}^{\alpha} &= \sum_n q(z_n = k)\big\{\log \phi_n^{(m)} - \big[\psi\big(\alpha_k^{(m)}\big) - \psi\big(\alpha_k^{(m)} + \beta_k^{(m)}\big)\big]\big\} \\ g_{\lambda_k}^{\beta} &= \sum_n q(z_n = k)\big\{\log\big(1 - \phi_n^{(m)}\big) - \psi\big(\beta_k^{(m)}\big) - \psi\big(\alpha_k^{(m)} + \beta_k^{(m)}\big)\big]\big\}, \end{aligned} \tag{28}$$

and Hessian $\mathbf{H}_{\lambda_k}$,

$$\mathbf{H}_{\lambda_k} = \sum_n q(z_n = k) \cdot \begin{bmatrix} \psi'\big(\alpha_k^{(m)} + \beta_k^{(m)}\big) - \psi'\big(\alpha_k^{(m)}\big) & \psi'\big(\alpha_k^{(m)} + \beta_k^{(m)}\big) \\ \psi'\big(\alpha_k^{(m)} + \beta_k^{(m)}\big) & \psi'\big(\alpha_k^{(m)} + \beta_k^{(m)}\big) - \psi'\big(\beta_k^{(m)}\big) \end{bmatrix}, \tag{29}$$

where $\psi(\cdot)$ is the *digamma* function and $\psi'(\cdot)$ the *trigamma* function. We can see that the forms of Equation (28) and Equation (29) are the same as that in the chaining-based model (Equation (17) and Equation (18)); however, the posterior distribution of the class labels $q(z_n)$ used differs; here in the unified LCM, the class label posterior distribution $q(z_n)$ is computed based on the knowledge of the discriminative parameter $\mathbf{w}$ (Equation (25)).

Finally, the full EM algorithm proceeds as follows. First, we initialize $\{\boldsymbol{\lambda}_k^{(m)}\}$ with uninformative prior $[1, 1]$ and $\mathbf{w}$ randomly. Then the algorithm is performed iteratively by alternatively executing the *E-step* and *M-step* in each iteration until the log-likelihood $\log p(\boldsymbol{\Phi}, \mathbf{Z}|\mathbf{w}, \boldsymbol{\lambda})$ converges.

5.4. Collusion Inference and Prediction

After performing model learning and inference for LCM (both the chaining-based model and the unified model), we can finally solve Problem 1.

For the collusion inference task, in which the goal is to infer the class labels of the candidate groups given by the data, we can obtain the probability of each group g_n

being a colluder group ($p(z_n = 1)$) or a non-colluder group ($p(z_n = 0)$) by computing the posterior distribution over its class label z_n, which is readily given by Equation (14) and Equation (25):

$$\begin{aligned} p(z_n = 1) &= q(z_n = 1|\boldsymbol{\phi}_n, \mathbf{w}^*, \boldsymbol{\lambda}^*) = \sigma(\mathbf{w}^{*T}\boldsymbol{\phi}_n + \triangle_{\lambda^*}) \\ p(z_n = 0) &= q(z_n = 0|\boldsymbol{\phi}_n, \mathbf{w}^*, \boldsymbol{\lambda}^*) = 1 - q(z_n = 1|\boldsymbol{\phi}_n, \mathbf{w}^*, \boldsymbol{\lambda}^*), \end{aligned} \tag{30}$$

where $\mathbf{w}^*$ and $\boldsymbol{\lambda}^*$ are the (local) optimal solution obtained after the EM algorithm converges.

For the collusion prediction task, based on $\mathbf{w}^*$ and Equation (12), the collusion predictive distribution can be derived as

$$\begin{aligned} p(\hat{z} = 1|\hat{\boldsymbol{\phi}}, \boldsymbol{\Phi}, \boldsymbol{Z}) &= \int p(\hat{z} = 1|\hat{\boldsymbol{\phi}}, \mathbf{w}) p(\mathbf{w}|\boldsymbol{\Phi}, \boldsymbol{Z}) \mathrm{d}\mathbf{w} \\ &\approx \int p(\hat{z} = 1|\hat{\boldsymbol{\phi}}, \mathbf{w}^*) p(\mathbf{w}|\boldsymbol{\Phi}, \boldsymbol{Z}) \mathrm{d}\mathbf{w} \\ &= p(\hat{z} = 1|\hat{\boldsymbol{\phi}}, \mathbf{w}^*) = \sigma(\mathbf{w}^{*T}\boldsymbol{\phi}_n). \end{aligned} \tag{31}$$

6. EXPERIMENTAL ANALYSIS

In this section, we conduct a series of experiments based on two real-world datasets collected from popular review sites to evaluate the predictive and inference performance of the proposed LCM and compare with state-of-the-art collusion detection algorithms.

6.1. Datasets

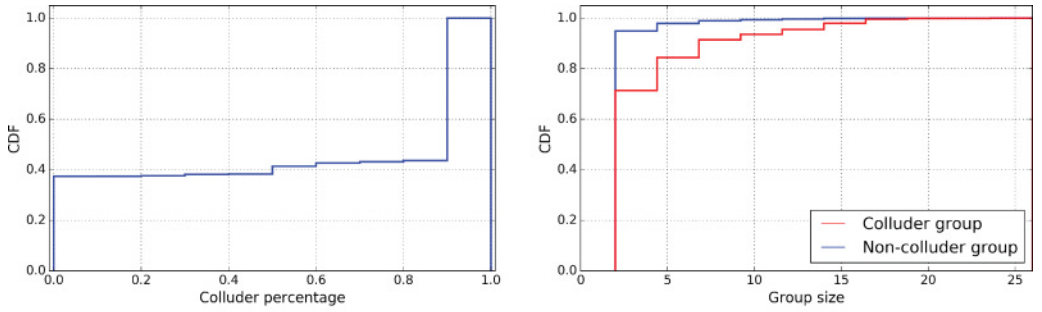
Our experiments are conducted on two golden standard datasets that contain collusive opinion fraud. As it is often very challenging to identify opinion fraud in general, the annotations of both datasets rely largely on the results provided by the anti-spam systems of the original sites. These systems are typically maintained by a team of domain experts with substantial anti-spam experience and are shown to be effective to defend against opinion fraud in online reviews [18].

6.1.1. Amazon Data. The first dataset, named AmazonCn, was created in our previous work [27], which contains consumer reviews about manufacture products sold on Amazon.cn. By following the same procedure and settings (i.e., minimum number of items reviewed by a group is 3 and minimum group size is 2^6) as in Reference [17], a total of 8,915 candidate colluder groups are generated. These groups are further labeled as colluder or non-colluder groups, and the users as colluders or non-colluders, based on two steps. First, provided that Amazon would periodically delete reviews if they are regarded as spam or their contents are irrelevant to the specific products, the users who have *at least one review* deleted are first treated as colluders.[7] Second, we label the candidate groups. Specifically, a group is regarded as a *colluder group* if it consists of colluders only; otherwise, it is labeled a *non-colluder group*.[8] It worth noting that it is possible that some mixed groups (i.e., consisting of both colluders and non-colluders) can actually be true colluder groups if the involved non-colluders have been mislabeled already. However, it is very challenging in this case to tell whether a mixed group is a true colluder group or not, as there is no ground-truth information in nature. It would thus be safer to regard groups containing only colluders as the positives (i.e., colluder groups),

[6]Note that this is the smallest size a group can have.

[7]This is based on the fact that (1) they are members of groups that have co-reviewed multiple items and (2) they are general spammers as officially discovered by the site.

[8]Here we implicitly assume that the labeled colluders (those officially identified by the site) in the same colluder groups are the members of the same campaigns, since they are targeting on the same items.



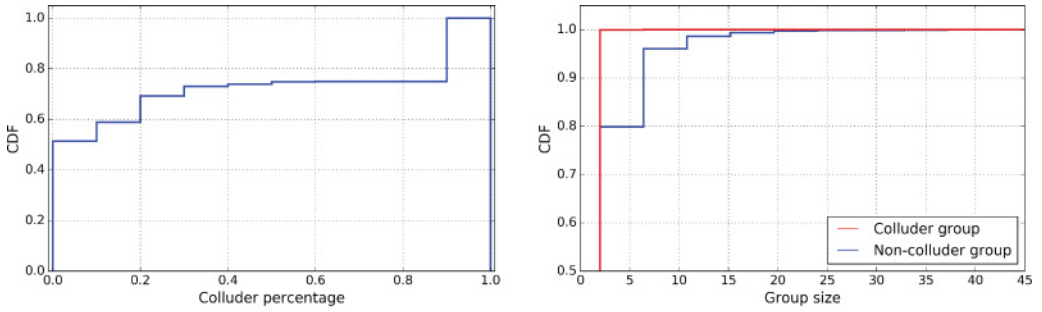
(a) Colluder percentage distribution (b) Group size distribution

Fig. 2. Some distributions of candidate colluder groups on the AmazonCn dataset.

such that the positive instances can be as accurate (pure) as possible. This can also be justified empirically by drawing the distribution of colluder percentage of a group, which is shown in Figure 2(a). It can be seen that most of the groups are pure, composed of either all colluders (56.14%) or all non-colluders (37.37%). The mixed groups are practically in the minority (6.49%), suggesting that empirically the impact from those possibly mislabeled true colluder groups mentioned before would be relatively minor. To further compare the two kinds of groups, Figure 2(b) shows the cumulative distributions of the size of colluder groups vs. non-colluder groups. We can see that there is a clear gap between the two distributions, with colluder groups having a relatively larger size.

In addition, as we find that not all spam reviews are removed from the system, manual annotation is also introduced to refine the initial annotations. Identifying suspicious spam reviews is a very challenging process wthat involves considerable effort on user behavior analysis via searching, collecting, and comparing related reviewing histories. However, the identification of spam reviews posted by colluders might be relatively easier, since one can compare the reviews posted by the group members. The manual annotation process is thus carefully conducted and strictly guided by a list of guidelines, which specify signals that are considered to be closely related to collusive fraud. These guidelines are desirable, since some of them have been used in previous studies for the same kind of annotating task [9, 17, 24], and others are obtained from various online resources compiled by experts with substantial experiences on opinion fraud detection. Some examples are listed as follows:

- *A group is suspicious if the sentiments of the reviews posted by the group members towards most of their common products are the same*. While it is reasonable for a group of users to either like or dislike one or two common products at the same time, it can be problematic if they agree on the same products almost all the time.
- *A group is suspicious if the texts of the reviews posted by the group members for most of their common products are similar*. For example, if multiple users always use lexically or semantically similar words or phrases to express their opinions towards the same products, it is very likely that they are instructed to do so as a group.
- *A group is suspicious if each of the group members always gives either positive or negative ratings*. It has been revealed that the ratings given by general spammers are often extreme and highly skewed [9, 16]. If the members of a candidate group all exhibit such a pattern, then this group is very likely to be a colluder group.
- *A group is suspicious if each of the group members posts reviews at concentrative times*. Posting reviews within short time intervals is another trait of general spammers [6, 27]. One reason is that, after being created, the spammer accounts are



(a) Colluder percentage distribution (b) Group size distribution

Fig. 3. Some distributions of candidate colluder groups on the YelpZip dataset.

Table I. Summary of the Datasets Used in the Experiments

Dataset	#reviews	#users	#items	#colluders	#non-colluders	mean group size
AmazonCn	1,205,125	645,072	136,785	1,937	3,118	3.54
YelpZip	608,598	260,277	5,044	459	2,580	4.14

found to be used mostly for conducting a series of spamming tasks subsequently. If the group members all have such an aggregative usage pattern, then the entire group can be suspicious.

Due to the limited information that is publicly available on various sites (e.g., text, rating, and timing), it is very hard to design guidelines that capture completely different signals from those used by the features. In our case, most of the signals in the guidelines are orthogonal (or at least used differently) to those captured by the proposed h-CBMs. More details about the annotation process can be found in Reference [27].

6.1.2. Yelp Data. The second dataset, named YelpZip, was created by the authors in Reference [22], which contains reviews about restaurants and hotels on Yelp.com. The original dataset contains annotations about general fake reviews and review spammers (based on the results of Yelp's own anti-spam system) but does not specify whether a user is a colluder or not. To obtain such information, we generate candidate groups from the original annotated datasets by following the same procedure as before and obtain 5,980 groups in total. Then, the labeling will proceed over all groups and the members of these groups. As with the case of the AmazonCn dataset annotation, a group is labeled as a colluder group if all its members are annotated as colluders in the original dataset. Figure 3 demonstrates the colluder percentage distribution (Figure 3(a)) and the group size distribution (Figure 3(b)) of the candidate colluder groups on the YelpZip dataset. From Figure 3(a), we can observe a similar trend that most the of candidate groups are the pure ones. In terms of the group size distribution, it turns out that, differing from the AmazonCn case, colluder groups in YelpZip generally contain fewer members, with over 90% colluder groups having size 2. This might indicate that the collusive fraud practice is milder in YelpZip than in AmazonCn. Finally, we list the summary statistics of the two datasets in Table I.

6.2. Effectiveness of h-CBMs

In this section, we evaluate the effectiveness of the proposed h-CBMs that are designed to capture the behavioral connections between colluders. In particular, we take a classification evaluation approach to assess the discriminating ability of each h-CBM by training a classifier for each *individual* h-CBM on the datasets. The classifier we use

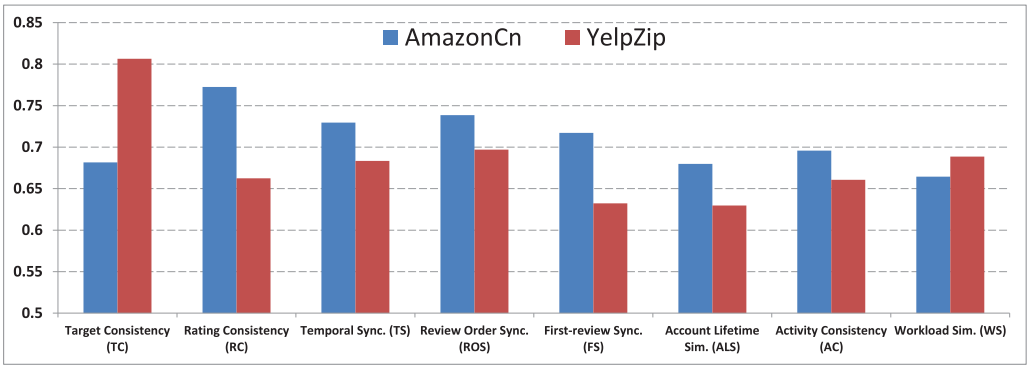


Fig. 4. Classification performance (AUC) of h-CBMs on AmazonCn and YelpZip.

here is Logistic Regression. The standard classification metric, Area under ROC curve (AUC), is employed. AUC score is commonly used to evaluate a binary classifier by inspecting the true positive rate against the false-positive rate at different discrimination thresholds. Five-fold cross-validation is used to conduct the experiment. The results are shown in Figure 4.

In general, it can be seen that all h-CBMs are useful in characterizing the collective behaviors of colluders, as all of them attain AUC $>$ 0.5 (0.629 precisely). On the AmazonCn dataset, the best result is achieved by rating consistency (RC), implying that a strong agreement is reached among the colluders on the sentimental disposal of the targets in a particular fraud campaign. Temporal-based h-CBMs (i.e., temporal synchronization (TS), review order synchronization (ROS), and first-review synchronization (FS)) also works well, which explains the burstiness of typical review fraud campaigns; the involved colluders are coordinated and thus synchronized, as each one of them is a part of the collaboration. Finally, activity-based h-CBMs (i.e., account lifetime similarity (ALS), workload similarity (WS), and activity consistency (AC)) show their strengths by capturing the resemblance of the account usage patterns of colluders. In particular, the promising result achieved by activity consistency (AC) suggests a new angle to quantify the connections between colluders by relating their most active moments.

On the YelpZip dataset, we can observe a different behavioral distribution. Here the best performance is achieved by target consistency (TC), showing that the colluders' accounts on Yelp are used almost exclusively for the designated targets. Again, rating- and temporal-based h-CBMs outperform activity-based h-CBMs, suggesting that colluders' actions are typically bounded by the campaigns they engage in, such that the similarity between their actions in terms of rating and time would become more distinctive. Finally, we see that the h-CBMs generally perform better on Amazon than on Yelp. This might indicate that the fraud campaigns on Yelp are organized in a looser manner in the sense that the desired fake reviews are spread across a longer period of time, and the usage patterns of the involved accounts are more diverse. In addition, this difference also shows that the behaviors of colluders differ substantially in different domains (i.e., manufacture products vs. restaurants). Nonetheless, as being members of fraud campaigns, colluders would necessarily establish certain behavioral connections incurred due to the common campaign goals and activities, and this behavior can effectively be captured by the proposed h-CBMs.

6.3. Prediction Performance

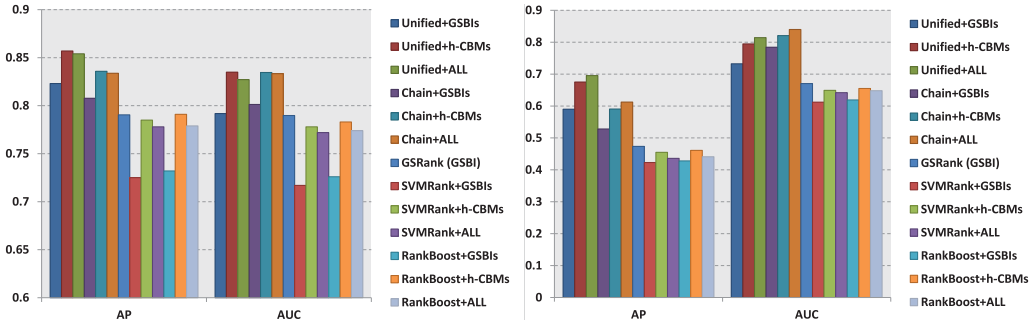
In this section, we evaluate the predictive performance of LCM and make a comparison with state-of-the-art *unsupervised* collusive fraud detectors. As the output of LCM is

a list of probabilities of being colluders for each user in the dataset, for an evaluation protocol, we employ two well-known ranking-based metrics, namely Average Precision (AP) and Area AUC. In summary, we have the following baselines for the comparison analysis:

- GSRank: This is an unsupervised ranking algorithm [17] proposed to detect colluder groups. It works by performing iterative computations on three related entities, that is, users, user groups, and products. In addition, eight Group Spam Behavior Indicators (GSBIs) are proposed to capture suspicious group behaviors of colluders. In particular, GSBIs include features that investigate the *external* properties of a candidate group, such as *Group Size* and *Group Support Count* mentioned in Section 3 that produce measurements of a whole group. Differing from GSBIs, the proposed h-CBMs in this work focus on the *internal* properties of a candidate group. Thus, even though a colluder group can restrict its size or overwhelm the targets, the actions of its members towards the targets are inevitably homogeneous, which renders it detectable. For the experiments to be performed, as GSRank is tightly coupled with the GSBIs and cannot freely incorporate other features, we will evaluate the performance of GSRank combined with GSBIs only.
- Learning to rank: Another competitor capable of unsupervised training is the learning-to-rank approach. This approach is traditionally used in supervised learning to learn an optimal combination of a list of training rankings. It, however, can be adapted to detect opinion fraud in unsupervised settings through proper extension [16, 17]. The insight is that when the value of a feature is semantically meaningful (for example, the larger the value, the more suspicious the sample), a valid training ranking can be generated by sorting the samples based on their feature values in descending order. In our case, we obtain eight training rankings, each of which corresponds to one dimension of the h-CBM vectors. Two well-known learning-to-rank algorithms, SVMRank [10] and RankBoost [7], are included as baselines. In the experiments, the two algorithms will be performed with GSBIs, h-CBMs, and both.
- LCM Chain (Chain for short): For the chain-based LCM model, we use Beta mixture model for collusion inference and Logistic Regression for collusion prediction, respectively. As a flexible model that can work with any number of features, Chain will be performed with GSBIs, h-CBMs, and both.
- LCM Unified (Unified for short): Aimilarly to the LCM Chain model, LCM Unified will be performed with GSBIs, h-CBMs, and both.

The output of each baseline is a score assigned to each group representing the possibility of being a colluder group. In LCM, the score for each group is $p(\hat{z}|\hat{\phi}, \mathbf{\Phi}, \mathbf{Z})$, as given by Equation (31). The experiment is conducted using fivefold cross-validation. For GSRank, as it is not a trainable model, the evaluation is performed based on one test folder each time. Our implementations of the EM algorithms for LCM converge in less than 18 iterations given that the algorithms terminate when the likelihood difference between consecutive iterations falls below 10^{-6}. The results are shown in Figure 5. In general, we can see that the proposed LCM (both Chain and Unified) achieves better performance than the second-place GSRank in both metrics on the two datasets. For LCM, the two variants, that is, Chain and Unified, are observed to compliment each other when different metrics are used.

On the AmazonCn dataset, as shown in Figure 5(a), it turns out that Unified generally outperforms Chain, with the best result achieved by using h-CBMs, that is, for Unified+h-CBMs, we have AP = 0.857 and AUC = 0.835. For the feature performance comparison, we inspect the experiments that use the same models. It shows that in all cases the performance can be improved by incorporating h-CBMs for detection (by comparing X+GSBIs with X+ALL, with "X" representing the model used). The



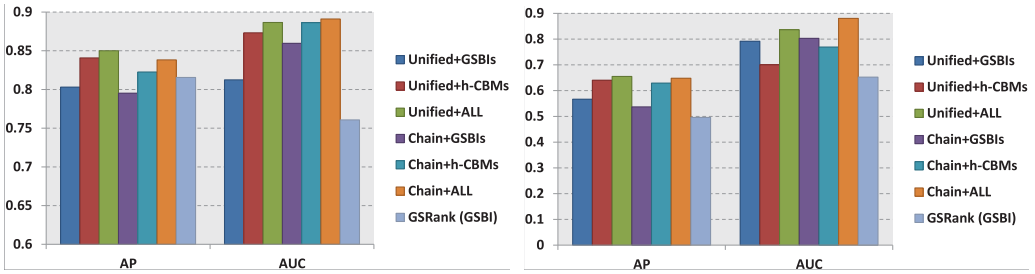
(a) Amazon (b) Yelp

Fig. 5. Collusion prediction performance (in Average Precision and AUC) of LCM compared with GSRank on the AmazonCn and YelpZip datasets.

improvements range from 3.3% to 7.3% in AP and 3.9% to 7.6% in AUC. Also, the h-CBMs are shown to perform better than GSBIs (by comparing X+GSBIs with X+h-CBMs), by a margin of 3.5%-8.2% in AP and 4.1%-8.5% in AUC, which suggests the effectiveness of characterizing colluders' homogeneous behaviors in practice. Then, to compare the performance of different models, we inspect the experiments with the feature set fixed (i.e., X+GSBIs). It can be seen that the proposed LCM (both Unified and Chain) exhibits superior performance over the second-place GSRank, with the improvements by 2.3%–4.1% in AP and 0.3%–1.5% in AUC.

On the YelpZip dataset, as shown in Figure 5(b), we can observe similar trends in terms of model and feature improvements. In particular, for the feature comparison, we compare the experiments X+h-CBMs and X+GSBIs with "X" being LCM Chain, LCM Unified, SVMRank, and RankBoost. It can be seen that h-CBMs generally achieve better results than GSBIs, with the improvements ranging from 7.6% to 14.4% in AP and 4.7% to 8.5% in AUC. For the model comparison, we see that the performance of GSRank+GSBIs is much worse than its performance on the AmazonCn dataset. This might be ascribed to the fact that GSRank, as a power iteration algorithm, can amply errors during propagation, which would dramatically deteriorate the performance when the used features are incapable of capturing the behaviors of colluders. In contrast, LCM is more robust to less-effective features by modeling the joint probability of the class labels and feature values of candidate groups.

Finally, we compare the two LCM variants, which are observed to compliment each other in terms of ranking (AP) and classification (AUC) performance. Specifically, in terms of ranking, by fixing feature sets (i.e., comparison between X+GSBIs, X+h-CBMs, and X+ALL, respectively, with "X" being Unified or Chain), we can see that Unified outperforms Chain on both datasets with a margin from 1.9% to 2.5% on AmazonCn and 11.7% to 14.4% on YelpZip in AP. However, in terms of AUC score, the superiority is reversed. Since AUC considers true negatives (i.e., true normal users) when computing the false-positive rate and both of the datasets are slightly imbalanced with many more negatives (normal users) than positives (colluders), the superiority of Chain over Unified in AUC might indicate that Chain is more suitable for an imbalance classification task in which both colluders and normal users are of interest. This can be partially due to the fact that in Unified, the probabilities of the latent group class labels and the predictive weights are jointly modeled, which would cause the data used in the model to account for the estimation of both objects at the same time. In contrast, the data in Chain are fully used for the collusion inference task, which can lead to more accuracy results, since more data are exploited to estimate the group class labels alone.



(a) Amazon (b) Yelp

Fig. 6. Collusion inference performance (in Average Precision and AUC) of LCM compared with GSRank on AmazonCn and YelpZip datasets.

6.4. Inference Performance

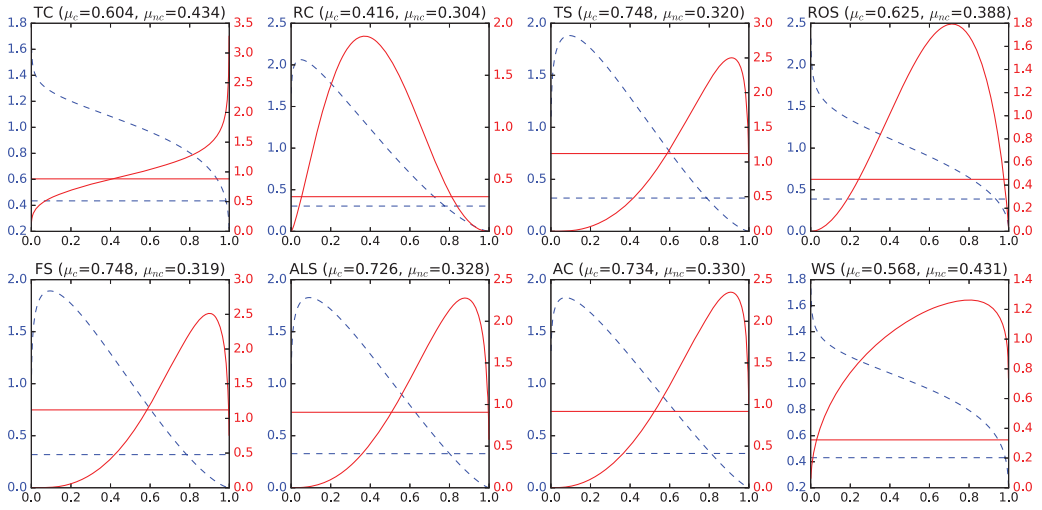
In this section, we evaluate the collusion inference performance of the proposed LCM for collusive fraud detection. Note that in LCM, the inference task is performed by computing the posterior distribution of the group class labels according to Equation (30). Differing from the collusion prediction task discussed before, here the experiment can be conducted and evaluated on the full datasets. Note that the learning-to-rank algorithms (i.e., SVMRank and RankBoost) cannot perform the collusion inference task, as they are supervised models, and thus they are not included in this comparison. Figure 6 shows the comparison results between LCM and GSRank in AP and AUC on the two evaluated datasets.

The first observation we make is that on the AmazonCn dataset, where a number of large colluder groups exist, by utilizing the full dataset for inference, GSRank improves its performance by attaining a slightly higher AP (0.815) than LCM+GSBIs (i.e., Chain+GSBIs and Unified+GSBIs). However, the AUC score of GSRank turns out to be significantly lower, suggesting that GSRank still cannot properly distinguish all colluders from non-colluders on the AmazonCn dataset. This can be ascribed to the fact that, apart from large groups, there are also a non-trivial number of small colluder groups whose behaviors are stealthier and cannot be effectively captured by the GSBIs that look at the external properties of candidate groups (e.g., group size and target set size) that are more vulnerable to manipulation. In contrast, the performance of LCM Chain and Unified achieve better AUC scores, with the help of the h-CBMs that capture the similarity between colluders rather than their aggregated characteristics.

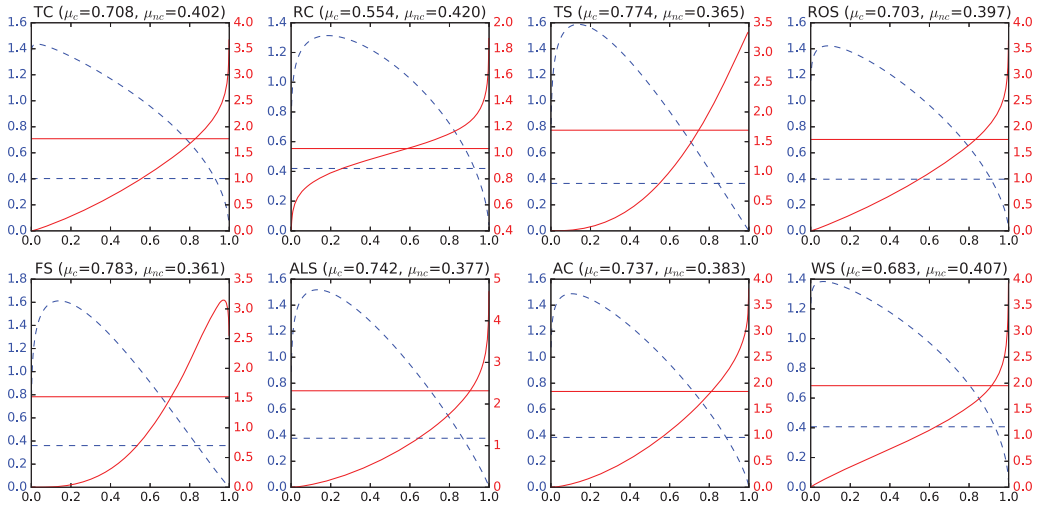
Second, by comparing the inference performance of the two variants of LCM, similar trends are observed in terms of their ranking and classification abilities as those in the collusion prediction comparison. Again, on both datasets, LCM Unified achieves higher AP score than Chain when working with each of the evaluated feature sets, with the improvements ranging from 1.4% to 2.2% on AmazonCn and 1.0% to 5.6% on YelpZip. On the other hand, in terms of AUC performance, LCM Chain dominates the results on both datasets. This again, as discussed in the previous subsection, shows that LCM Chain would be more suitable for a binary classification task on imbalanced datasets where both colluders and non-colluders are of interest, while LCM Unified would be appropriate for ranking-based retrieval scenarios where the goal is to rank colluders as highly as possible in the obtained results.

6.5. Learned Collusive Behavioral Patterns

In addition to performing collusion prediction and inference tasks, LCM can also learn the behavior patterns of colluders and non-colluders, which are modeled as Beta distributions introduced in the generative view of LCM (Section 5.1.1). As mentioned,



(a) LCM Unified on AmazonCn dataset



(b) LCM Unified on YelpZip dataset

Fig. 7. The learned collusive behavioral patterns on AmazonCn and YelpZip datasets. Each behavioral pattern is modeled as a Beta density function (PDF) and corresponds to each h-CBM dimension $\phi \in \{\text{TC, RC, TS, ROS, FS, ALS, AC, WS}\}$. The density functions for colluders and non-colludes are shown in red/solid lines and blue/dotted lines, respectively. The expected values for each learned behavioral distributions are plotted as horizontal lines and also reported in the plot captions.

the behavioral (Beta) distributions of colluders and non-colluders are parametrized by the Beta parameters $\boldsymbol{\lambda}_1 = \{\lambda_1^{(m)}\}$ and $\boldsymbol{\lambda}_0 = \{\lambda_0^{(m)}\}$, respectively, where each $(\lambda_1^{(m)}, \lambda_0^{(m)})$ pair corresponds to the distributions of the mth h-CBM dimension over colluder and non-colluder clusters, respectively.

Figure 7 shows the behavioral distributions learned by LCM Unified for colluders and non-colluders on the two datasets. We omit the results of LCM Chain as the learned distributions are similar. In Figure 7, each sub-plot shows the learned two

Beta distributions parametrized by $\boldsymbol{\lambda}_1^{(m)} = \{\alpha_1^{(m)}, \beta_1^{(m)}\}$ and $\boldsymbol{\lambda}_0^{(m)} = \{\alpha_0^{(m)}, \beta_0^{(m)}\}$ on the mth observed h-CBM dimension $\phi_{(m)} \in \{$TC, RC, TS, ROS, FS, ALS, AC, WS$\}$. In particular, the distributions for colluders and non-colluders are shown in red (solid) lines and blue (dotted) lines. The expected values of the distributions are shown as horizontal lines. The x-axis is the domain of Beta distribution that is also the range of the corresponding h-CBMs, with values close to 1 indicating more likely to be related to collusive fraud.

In general, from Figure 7 we observe clear deviations between the learned behavioral distributions of colluders and non-colluders in all sub-plots, where the densities of colluders put more mass towards the extreme right of each sub-plot, showing that the behaviors of colluders are more closely related to collusive fraud that is characterized by the proposed h-CBMs. This not only validates the hypothesis that the behaviors of colluders and non-colluders differ and can be separable in appropriate feature space but also shows the effectiveness of LCM in learning the separated distributions. In addition, the deviation degrees of the two distributions (i.e., for colluders and non-colluders) can also to some extent reflect the relative strengths of the features used. For example, on the AmazonCn dataset in Figure 7(a), we see that the distribution gaps are larger on temporal synchronization (TS), first-review (FS), account lifetime similarity (ALS), and activity consistency (AC) dimensions than the rest, which is generally consistent with the results of h-CBM evaluation in Section 6.2. Similar observations are made on target consistency (TC) and rating consistency (RC) dimensions for the YelpZip dataset. It is worth noting that the results here do not perfectly match those in h-CBM evaluation, because in Section 6.2 each h-CBM is evaluated independently, while the shown behavioral parameters $\boldsymbol{\lambda}$ here are learned jointly in LCM such that a balance is made over all h-CBM dimensions to obtain the maximum model likelihood and optimal group class label posteriors.

7. RELATED WORK

In this section, we survey several lines of research that are mostly related to our work on collusive opinion fraud detection in online reviews.

7.1. Opinion Fraud Detection

The studied problem here is within the broader topic on opinion fraud detection, where the goal is to identify deceptive reviews in a review portal that are created for manipulating the online reputation of items (e.g., products or restaurants). There has been a large body of work on this topic that typically formulates this detection task as a binary classification problem and relies on machine-learning techniques to perform this task. The goal of detection then becomes the classification of a particular review as fake (positive) or not (negative). At the beginning, supervised learning approaches have commonly been used for this classification task [9, 20]. These approaches require annotated fake review data and proceed commonly in two steps. First, a set of features are extracted from all the information regarding a review-posting event. Then, predictive models are trained on the annotated dataset based on those engineered features. Once the training is finished, the trained models can finally be applied to fake review classification/detection.

Jindal and Liu [9] identify three types of spam reviews on a popular product review portal that includes deceptive reviews, non-specific reviews (reviews not on the specific items), and non-reviews (e.g., ads). They extract more than 30 features that are engineered from three aspects of review postings, including review-centric features (e.g, review text, rating, and length), user-centric features (e.g., average rating given and standard deviation in rating), and product-centric features (e.g., average rating

received, price, and sales rank). For the classification model training, logistic regression is utilized with various combinations of the features and achieves promising results when all features are used. Ott et al. [20] study further the effectiveness of review text features in fake review detection. In this work, the problem of fake review detection is formulated as a text categorization task where a classification model is trained on the text of a review (e.g., n-grams). In addition, a set of features that are traditionally used in psychological studies of deception and genre identification are also evaluated on this task and are found to achieve further improvements.

However, the major problem of these predictive models is that they hinge heavily on the annotation of fake reviews. This information is often difficult to obtain due to the lack of *ground truth* in reality. To overcome this problem, several inference-based approaches have been proposed to operate in unsupervised settings [1, 16, 17, 25]. The basic idea is to generate a numerical score for each entity of interest (i.e., a review or a user) to represent its suspicious level and then order them in a ranked list. The higher an entity ranks, the more suspicious it is.

In Wang et al. [25], the score is generated through an iterative computation framework where specific functions are proposed to model the interplay between three related concepts, that is, trustiness of users, honesty of reviews, and reliability of items. The underlying hypothesis is that the honesty of a review can be judged by the reliability of the item for which it is written. In addition, the trustworthy of a user can be inferred from the honesty of her reviews. Finally, the reliability of an item can be told through inspecting the review sentiments (i.e., positive or negative) of trustworthy users who post reviews for it. Similar ideas are also found in the work of Reference [1], where a Markov random field is used to model the relationships among reviews, users, and items. In Mukherjee et al. [16], a generative model is proposed to model the process that generates review features or user features from latent class variables. However, the generative process is designed for modeling individual spammers rather than colluder groups.

The work in Reference [17] is the most related to ours where a power-iteration based framework is proposed to detect colluder groups. A set of GSBIs are proposed to score the suspiciousness of colluders, items, and colluder groups. Based on these scores, an HITS-like power-iteration algorithm is proposed to generate a final score for each involved entity.

However, a major problem of the inference-based approaches is that they are not designed to learn from data for making predictions. Given a set of review history data as input, they can only detect the occurred fraud activities in the provided data. No models are learned for predictive purposes. In contrast, the proposed approach in this article tackles collusive opinion fraud from a probabilistic perspective and can perform both collusion inference and prediction in a principled manner.

7.2. Features for Characterizing Opinion Fraud

In the literature, for both supervised and unsupervised scenarios, a large number of features have been used to characterize opinion fraud on various dimensions, and the qualities of the features are vital to the ultimate detection performance. In general, a set of linguistic and behavioral features are commonly used to capture the characteristics of fake reviews and fraudsters [9, 13, 16, 20]. For example, n-grams of review text in proposed in Reference [20] are used to capture the wording patterns of fake reviews on vocabulary and phrase level. The review text similarity proposed in References [9, 13] is based on the observation that fraudsters tend to copy or slightly modify existing fake reviews for time saving. In Reference [13], deviation-based features are proposed to evaluate how a particular user behaves differently from others in terms of giving

ratings and review posting timing. However, these features can only work on individual users and are not designed for colluders who commit collusive fraud.

For colluder behavior characterization, Mukherjee et al. [17] propose GSBIs. Similarly to the proposed h-CBMs in this article, these features focus on the collective behaviors of colluders and measure the properties of colluders as a group. However, most of the features in GSBIs are holistic oriented in the sense that they only investigate the *external* properties of a group. For example, *Group Size* and *Group Support Count* evaluate the size of a group and the number of items reviewed by a group, respectively. In the original article, a larger feature value is considered more suspicious in collusive fraud. As such, these features would only be suitable for finding large-scale collusion activities and cannot deal with stealthier attacks that involve small groups. Another feature type in GSBIs focuses on behavioral deviations of colluders from the general public. For example, *Group Deviation* and *Group Size Ratio* regard colluders as anomaly and evaluate how much their behaviors deviate from other users in terms of average ratings and size ratio. However, by leveraging the inherent advantage in manpower, colluders could easily turn themselves into the majority by overwhelming the targeted items, such that the deviations can be completely eliminated. The proposed h-CBMs on the other hand can handle such issues. By investigating the *internal* properties of colluder groups and modeling the homogeneity in colluders' collective behaviors, even though they can launch small-scale attacks or dominate the targeted items, the actions needed for the fraud tasks are inevitably homogeneous and can thus be captured by h-CBMs.

7.3. Collusive Fraud on the Web

Apart from online review domains, collusive fraud has also been unveiled in other applications. For example, in social networks, Beutel et al. [2] defend against collusive fraud in Facebook Page Likes. They represent a Page Like dataset as a bipartite graph that links the pages and the users who Like them. Then an iterative optimization algorithm is implemented to infer synchronized group attackers who co-Like a non-trivial number of pages in a short time period. Yang et al. [28] implement a propagation-based algorithm to infer malicious user accounts in Twitter where some criminal accounts are found to cover for each other through mutual endorsements. Another propagation-based approach is also proposed to identify collusive link farming in Twitter [8]. However, as no friendship links exist among users in most review sites, the work of References [28] and [8] cannot directly apply to the context of online reviews. Moreover, all the aforementioned approaches still suffer from the same issue as the inference-based fake review detectors that they cannot learn from existing data for predictive purposes. Finally, these approaches are not designed to work freely with other useful collusion features as the feature functions are coupled with the designed algorithms.

8. CONCLUSIONS

In this article, we identify the problem of detecting collusive fraud in online reviews from a stochastic perspective. In particular, we propose a novel statistical model called LCM to model collusive opinion fraud from a hybrid of generative and discriminative perspectives. Not only can LCM perform collusion inference as unsupervised models, but it can also make collusion predictions as supervised models. Two variants of LCM are studied in detail. The first is a chaining-based model that performs collusion inference and prediction in a sequential manner. The chaining-based model is flexible in the sense that models suitable for the respective tasks can be optimized and applied separately. Moreover, the experiment results show that it works empirically well on classifying colluder and non-colluders in scenarios where data are imbalanced. The

second is a unified model that combines the learning and inference of the generative and discriminative views to achieve optimal results via balancing between the two views. The experiment results show its superiority over the chaining-based model in colluder ranking ability. Furthermore, a suite of homogeneity-based collusive behavior measures (h-CBMs) are proposed to capture the homophily inside colluder groups. The h-CBMs are complementary to existing collusion-oriented features in handling stealthier collusive attacks. In summary, our results on two real-world datasets show the effectiveness of h-CBMs and the superiority of LCM over state-of-the-art competitors in terms of collusion inference and predictive abilities. The source code and datasets used in this article can be found at https://sites.google.com/site/homecxu/.

ACKNOWLEDGMENTS

We gratefully thank the reviewers for their constructive and valuable comments and thank the reviewers of the preliminary version that appeared at IEEE ICDM15.

REFERENCES

[1] L. Akoglu, R. Chandy, and C. Faloutsos. 2013. Opinion fraud detection in online reviews by network effects. In *Proceedings of the 7th International AAAI Conference on Web and Social Media (ICWSM'13)*. 2–11.

[2] Alex Beutel, Wanhong Xu, Venkatesan Guruswami, Christopher Palow, and Christos Faloutsos. 2013. CopyCatch: Stopping group attacks by spotting lockstep behavior in social networks. In *Proceedings of the 22nd International Conference on World Wide Web (WWW'13)*. 119–130.

[3] Christopher M. Bishop et al. 2006. *Pattern Recognition and Machine Learning*. Vol. 4. Springer, New York, NY.

[4] Chester I. Bliss. 1934. The method of probits. *Science* 79, 2037 (1934), 38–39.

[5] Arthur P. Dempster, Nan M. Laird, and Donald B. Rubin. 1977. Maximum likelihood from incomplete data via the EM algorithm. *J. Roy. Stat. Soc. Ser. B* 39, 1 (1977), 1–38.

[6] Geli Fei, Arjun Mukherjee, Bing Liu, Meichun Hsu, Malu Castellanos, and Riddhiman Ghosh. 2013. Exploiting burstiness in reviews for review spammer detection. In *Proceedings of the 7th International AAAI Conference on Web and Social Media (ICWSM'13)* 13 (2013), 175–184.

[7] Yoav Freund, Raj Iyer, Robert E. Schapire, and Yoram Singer. 2003. An efficient boosting algorithm for combining preferences. *J. Mach. Learn. Res.* 4, 11 (2003), 933–969.

[8] Saptarshi Ghosh, Bimal Viswanath, Farshad Kooti, Naveen Kumar Sharma, Gautam Korlam, Fabricio Benevenuto, Niloy Ganguly, and Krishna Phani Gummadi. 2012. Understanding and combating link farming in the twitter social network. In *Proceedings of the 21st International Conference on World Wide Web (WWW'12)*. 61–70.

[9] N. Jindal and B. Liu. 2008. Opinion spam and analysis. In *Proceedings of the 2008 International Conference on Web Search and Data Mining (WSDM'08)*. 219–230.

[10] Thorsten Joachims. 2002. Optimizing search engines using clickthrough data. In *Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'02)*. 133–142.

[11] A. Jordan. 2002. On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes. *Adv. Neur. Inform. Process. Syst.* 14 (2002), 841.

[12] Julia A. Lasserre, Christopher M. Bishop, and Thomas P. Minka. 2006. Principled hybrids of generative and discriminative models. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Vol. 1. IEEE, 87–94.

[13] E. P. Lim, V. A. Nguyen, N. Jindal, B. Liu, and H. W. Lauw. 2010. Detecting product review spammers using rating behaviors. In *Proceedings of the 19th ACM International Conference on Information and Knowledge Management (CIKM'10)*. 939–948.

[14] Michael Luca and Georgios Zervas. 2013. *Fake it Till You Make It: Reputation, Competition, and Yelp Review Fraud*. Harvard Business School NOM Unit Working Paper 14-006.

[15] Arash Molavi Kakhki, Chloe Kliman-Silver, and Alan Mislove. 2013. Iolaus: Securing online content rating systems. In *Proceedings of the 22nd International Conference on World Wide Web (WWW'13)*. 919–930.

[16] Arjun Mukherjee, Abhinav Kumar, Bing Liu, Junhui Wang, Meichun Hsu, Malu Castellanos, and Riddhiman Ghosh. 2013. Spotting opinion spammers using behavioral footprints. In *Proceedings of*

the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'13). 632–640.

[17] A. Mukherjee, B. Liu, and N. Glance. 2012. Spotting fake reviewer groups in consumer reviews. In *Proceedings of the 21st International Conference on World Wide Web (WWW'12)*. 191–200.

[18] Arjun Mukherjee, Vivek Venkataraman, Bing Liu, and Natalie S Glance. 2013. What yelp fake review filter might be doing?. In *Proceedings of the 7th International AAAI Conference on Web and Social Media (ICWSM'13)*.

[19] Myle Ott, Claire Cardie, and Jeff Hancock. 2012. Estimating the prevalence of deception in online review communities. In *Proceedings of the 21st International Conference on World Wide Web (WWW'12)*. 201–210.

[20] Myle Ott, Yejin Choi, Claire Cardie, and Jeffrey T. Hancock. 2011. Finding deceptive opinion spam by any stretch of the imagination. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, Volume 1*. 309–319.

[21] Mahmudur Rahman, Bogdan Carbunar, Jaime Ballesteros, George Burri, and Duen Horng Polo Chau. 2014. Turning the tide: Curbing deceptive yelp behaviors. In *Proceedings of the 2014 SIAM International Conference on Data Mining (SDM'14)*. 244–252.

[22] Shebuti Rayana and Leman Akoglu. 2015. Collective opinion spam detection: Bridging review networks and metadata. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 985–994.

[23] Huan Sun, Alex Morales, and Xifeng Yan. 2013. Synthetic review spamming and defense. In *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'13)*. 1088–1096.

[24] G. Wang, S. Xie, B. Liu, and S. Y. Philip. 2012a. Identify online store review spammers via social review graph. *ACM Trans. Intell. Syst. Technol.* 3, 4 (2012), 61.

[25] Guan Wang, Sihong Xie, Bing Liu, and Philip S. Yu. 2012b. Identify online store review spammers via social review graph. *ACM Trans. Intell. Syst. Technol.* 3, 4 (2012), 61.

[26] Chang Xu and Jie Zhang. 2015. Towards collusive fraud detection in online reviews. In *Proceedings of the 2015 IEEE International Conference on Data Mining (ICDM'15)*. 1051–1056.

[27] Chang Xu, Jie Zhang, Kuiyu Chang, and Chong Long. 2013. Uncovering collusive spammers in chinese review websites. In *Proceedings of the 22nd ACM International Conference on Information and Knowledge Management (CIKM'13)*. 979–988.

[28] Chao Yang, Robert Harkreader, Jialong Zhang, Seungwon Shin, and Guofei Gu. 2012. Analyzing spammers' social networks for fun and profit: A case study of cyber criminal ecosystem on twitter. In *Proceedings of the 21st International Conference on World Wide Web (WWW'12)*. 71–80.

[29] Junting Ye and Leman Akoglu. 2015. Discovering opinion spammer groups by network footprints. In *Machine Learning and Knowledge Discovery in Databases*. 267–282.

Received June 2016; revised March 2017; accepted May 2017