Collective Opinion Spam Detection: Bridging Review Networks and Metadata

Shebuti Rayana
Stony Brook University
Department of Computer Science
srayana@cs.stonybrook.edu

Leman Akoglu
Stony Brook University
Department of Computer Science
leman@cs.stonybrook.edu

1. INTRODUCTION

Online reviews are increasingly valuable resources for consumers to make decisions. They are powerful since they reflect testimonials of "real" people, unlike advertisements. Financial incentives associated with reviews, however, have created a market of (often paid) users to fabricate *fake reviews* to either unjustly hype or defame a product or business, the activities of whom are called *opinion spam* [4].

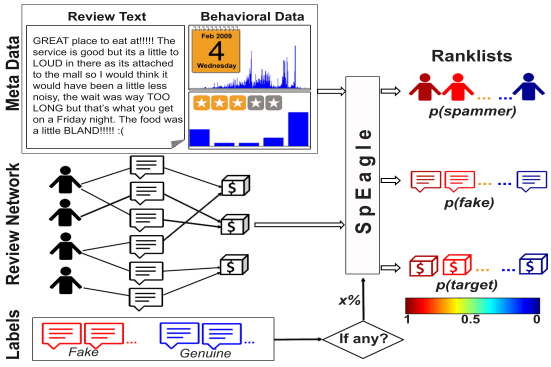


Figure 1: Workflow of SpEagle.

While the problem is surprisingly prevalent; it is a hard and mostly open problem. The key challenge is obtaining large ground truth data, however manual labeling of reviews is extremely difficult by merely reading them, where humans are only slightly better than random [11].

There have been considerable progress in opinion spam detection, however the problem remains far from fully solved. In this work, we capitalize on our prior work [1] to propose a new method, SpEagle (for Spam Eagle), that can utilize *metadata* (text, timestamp, rating) as well as *relational data* (review network) under a unified framework to spot spam users, fake reviews, as well as targeted products. We summarize the contributions of this work as follows.

- A new holistic approach SpEagle, which exploits both relational data (user–review–product graph) and metadata (text, timestamps, ratings) collectively to detect suspicious users, reviews, and targeted products.
- SpEagle is a review network-based classification task which accepts prior knowledge on the class distribution of the nodes (users, reviews, products) estimated from metadata by extracting indicative features of spam.
- SpEagle works in an unsupervised fashion, but can easily leverage labels (if available). As such, we introduce a *semi-supervised* version called SpEagle$^+$ which improves performance significantly without changing the inference steps of SpEagle.
- After investigating the effectiveness of all features, we design a *light* version of SpEagle called SpLite which uses a very small set of review features as prior information providing significant speed-up.

We evaluate our method on three real-world datasets collected from Yelp.com, containing filtered (spam) and recommended (non-spam) reviews. To the best of our knowledge, our work provides the largest scale *quantitative* evaluation to date for the opinion spam problem.

2. RELATED WORK

Opinion spam is one of the new forms of Web-based spam, and has been the focus of academic research in the last 7-8 years. Since the seminal work of Jindal *et al.* on opinion spam [4], a variety of approaches have been proposed. At a high level, those can be categorized as linguistic approaches [3, 11, 10] that analyze the language patterns of spam vs. benign users, for psycholinguistic clues of deception, behavioral approaches [4, 5, 8] that utilize the reviewing behaviors of users, (e.g., temporal and distributional footprints), and graph-based methods [1, 12, 6] that leverage the relation between users, reviews, and products with minimal to no external information. Our proposed approach utilizes clues from all of metadata (text, timestamp, rating) as well as relational data (network) , and harness them collectively under a unified framework to spot suspicious users and reviews, as well as targeted products of spam.

3. METHODOLOGY

In this work, we formulate the spam detection problem as a classification task on the user-review-product network. In this task, users are classified as *spammer* or *benign*, products as *targeted* or *non-targeted*, and reviews as *fake* or *genuine*. To aid the network classification, we utilize additional metadata (ratings, timestamps, and text) to extract indicative features of spam, which we incorporate into the inference procedure. Our proposed method works in an unsupervised fashion, however it can easily accommodate labels.

The network representation used by SpEagle is the user–review–product tripartite network. The network $G = (V, E)$ contains N user nodes $U = \{u_1, \ldots, u_N\}$, M product nodes $P = \{p_1, \ldots, p_M\}$, and Q review nodes $R = \{r_1, \ldots, r_Q\}$, $V = U \cup P \cup R$, connected through two types of edges the user-review edges $(u_i, r_k, t = \text{'}write\text{'}) \in E$ and the review-product edges $(r_k, p_j, t = \text{'}belong\text{'}) \in E$.

To formally define the classification problem, the network is represented as a pairwise Markov Random Field (MRF). The joint probability of node labels is written as a product

of individual and pairwise factors, parameterized over the nodes and the edges, respectively:

$$P(\mathbf{y}) = \frac{1}{Z} \prod_{Y_i \in V} \phi_i(y_i) \prod_{(Y_i, Y_j, t) \in E} \psi_{ij}^t(y_i, y_j) \quad (1)$$

where $\mathbf{y}$ denotes an assignment of labels to all nodes, y_i refers to node i's assigned label, and Z is the normalization constant. The individual factors ϕ_i are called *prior*, and represent initial class probabilities for each node. The pairwise factors ψ_{ij}^t are called *compatibility* (or edge) potentials, and capture the likelihood of a node with label y_i to be connected to a node with label y_j through an edge with type t. This is an inference problem which is combinatorially hard. Exact inference is known to be NP-hard for general MRFs, where instead iterative approximate inference algorithms such as Loopy Belief Propagation (LBP) [13] are used.

LBP is based on iterative message passing between the connected nodes. At every iteration, a *message* $m_{i \to j}$ is sent from each node i to each neighboring node j. The message captures the probability distribution over the class labels of j, and is computed as in Eqn.2,

$$m_{i \to j}(y_j) = \alpha \sum_{y_i \in \mathcal{L}_{T_i}} \phi_i(y_i)\ \psi_{ij}^t(y_i, y_j) \prod_{Y_k \in \mathcal{Y}_{\mathcal{N}_i} \setminus Y_j} m_{k \to i}(y_i) \quad (2)$$

where $\mathcal{N}_i$ denotes the set of i's neighbors, $T_i \in \{U, R, P\}$ denotes type of i and α is a normalization constant. These messages are exchanged iteratively over the edges until a "consensus" is reached. When the messages stabilize, we compute the marginal probability, called the *belief* $b_i(y_i)$, of assigning each Y_i associated with a node of type $T_i \in \{U, R, P\}$ with the label y_i in label domain $\mathcal{L}_{T_i}$ as follows,

$$b_i(y_i) = \beta\ \phi_i(y_i) \prod_{Y_j \in \mathcal{Y}_{\mathcal{N}_i}} m_{j \to i}(y_i) \quad (3)$$

where β is the normalization constant. For ranking, we sort by the probability values $b_i(y_i)$, where $y_i = spammer$ and $y_i = fake$ respectively for users and reviews.

In terms of setting the model parameters, we estimate the prior potentials ϕ_i from metadata and initialize (as follows) the compatibility potentials ψ_{ij}^t so as to enforce homophily.

Table 1: Compatibility potentials ψ^t used by SpEagle.

	User ($\psi^{t=\text{'write'}}$)		Product ($\psi^{t=\text{'belong'}}$)	
Review	*benign*	*spammer*	*non-target*	*target*
genuine	1	0	$1-\epsilon$	ϵ
fake	0	1	ϵ	$1-\epsilon$

To estimate the prior potentials ϕ_i, we first extract indicative features of spam from available metadata (ratings, timestamps, review text) for all three types of nodes and then convert them to prior class probabilities. Most of our features have been used several times in previous work on opinion spam detection, while several are introduced in this work. Table 2 includes brief descriptions for the features.

Given a set of values $\{x_{1i}, \ldots, x_{Fi}\}$ for the F features of a node i, we have to combine them into a spam score $S_i \in [0, 1]$, such that the class priors can be initialized as $\{1 - S_i, S_i\}$. To unify the features (having different scales) into a comparable interpretation, we leverage the cumulative distribution function (CDF). In particular, when we design the features, we have an understanding of whether a *high* (H) or a *low* (L) value is more suspicious for each feature. More formally, for each feature l, $1 \le l \le F$, and its corresponding value x_{li}, we compute

Table 2: Features for users, products, and reviews. H/L depicts if a High/Low value is spam.

User & Product Features			
behavior	MNR	H	Max. number of reviews written in a day [8, 9]
	PR	H	Ratio of positive reviews (4-5 star) [9]
	NR	H	Ratio of negative reviews (1-2 star) [9]
	avg/W RD	H	Avg./Weighted rating deviation [2, 7, 9]
	BST	H	Burstiness of reviews [2, 9]
	ERD	L	Entropy of rating distribution [new]
	ETG	L	Entropy of temporal gaps Δ_t's [new]
text	RL	L	Avg. review length in number of words [9]
	A/MCS	H	Avg./Max. content similarity [2, 7, 9]
Review Features			
behavior	Rank	L	Rank order among all the reviews of product [4]
	RD	H	Absolute rating deviation [5]
	EXT	H	Extremity of rating [8]
	DEV	H	Thresholded rating deviation of review [8]
	ETF	H	Early time frame [8]
	ISR	H	Is singleton? If review is user's sole review [new]
text	PCW	H	Percentage of ALL-capitals words [4, 5]
	PC	H	Percentage of capital letters [5]
	L	L	Review length in words [5]
	PP1	L	Ratio of 1st person pronouns ('I', 'my', etc.) [5]
	RES	H	Ratio of exclamation sentences containing '!' [5]
	SW	H	Ratio of subjective words (by sentiWordNet) [5]
	OW	L	Ratio of objective words (by sentiWordNet) [5]
	F	H	Frequency of review (approx. using LSH) [new]
	DL_u/DL_b	L	Description length based on uni/bi-grams [new]

$$f(x_{li}) = \begin{cases} 1 - P(X_l \le x_{li}), & \textit{if high is suspicious (H)} \\ P(X_l \le x_{li}), & \textit{otherwise (L)} \end{cases}$$

where X_l denotes a real-valued random variable associated with feature l with probability distribution P. Finally we combine these f values to compute the spam score of a node i as follows.

$$S_i = 1 - \sqrt{\frac{\sum_{l=1}^{F} f(x_{li})^2}{F}} \quad (4)$$

One of the key advantages of our formulation is that it enables seamless integration of labeled data when available. Specifically, given the labels for a set of nodes (reviews, users, and/or products), we simply initiate the priors as $\{\epsilon, 1-\epsilon\}$ for those that are associated with spam (i.e., *fake*, *spammer*, or *target*), and $\{1-\epsilon, \epsilon\}$ otherwise. The priors of unlabeled nodes are estimated from metadata as given in Eqn. (4). The inference procedure remains the same.

The original SpEagle computes all the features for every (unlabeled) node. In the experiments we investigate the effectiveness of the features and identify a small subset of review features that produces comparable performance to using all of them. As such, we propose a light version of our method, called SpLite (for SpEagle-Light), where we initialize the priors for unlabeled reviews based on the spam score computed only on those features, and use unbiased priors $\{0.5, 0.5\}$ for (unlabeled) users and products. This significantly reduces the feature extraction overhead, enabling speed-up with only slight compromise in performance.

4. EVALUATION

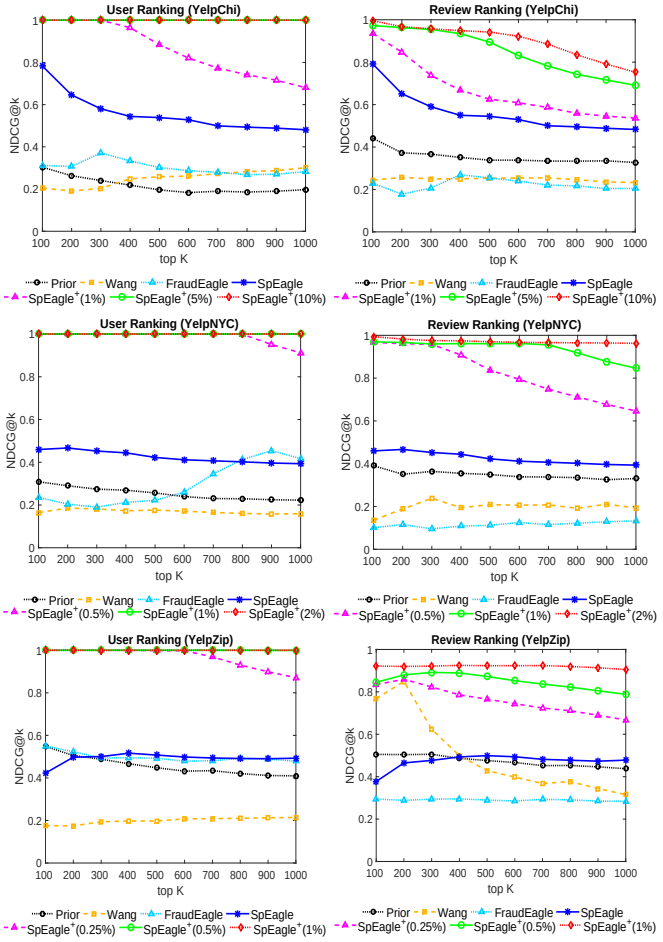
We evaluate our approach *quantitatively* on three real-world datasets (`YelpChi`, `YelpNYC` and `YelpZip`) collected from Yelp.com with near-ground-truth (recommended vs. filtered), summary statistics of which are given in Table 3.

We compare the performance of SpEagle to FraudEagle [1], a graph-based approach by [12] denoted as Wang

Table 3: Review datasets used in this work.

Dataset	#Reviews (filtered %)	#Users (spammer %)	#Products (rest.&hotel)
YelpChi	67,395 (13.23%)	38,063 (20.33%)	201
YelpNYC	359,052 (10.27%)	160,225 (17.79%)	923
YelpZip	608,598 (13.22%)	260,277 (23.91%)	5,044

ET AL., PRIOR(spam scores computed solely from metadata), semi-supervised SPEAGLE$^+$ with varying amount of labeled data, as well as to the computationally light version SPLITE.



(a) User Ranking (b) Review Ranking

Figure 2: $NDCG@k$ of compared methods.

Table 4 provides the AUC values over all three datasets for both user and review ranking. Notice that SPEAGLE outperforms FRAUDEAGLE, WANG ET AL. and PRIOR. The superiority of SPEAGLE's ranking becomes more evident when the top of the ranking results are considered through $NDCG@k$ in Figure 2. Next we analyze performance for varying amount of labeled data. Figure 2 shows the $NDCG@k$ performance of SPEAGLE$^+$, on all three datasets for both user and review ranking. We notice that the performance is improved considerably even with very small amount of supervision.

Moreover, our analyses suggest that (1) review priors alone are the most effective, and that (2) behavioral features are superior to text features. As feature extraction is expensive, our goal is to identify a few *behavioral* features for only the *review* nodes to be used in estimating priors fast to design a light version of SPEAGLE. We design SPLITE to utilize only two behavioral features for review nodes as estimated by our analysis for calculating priors. Figure 3 illustrates

Table 4: AUC performance of compared methods.

	User Ranking AUC			Review Ranking AUC		
	Y'Chi	Y'NYC	Y'Zip	Y'Chi	Y'NYC	Y'Zip
FRAUDEAGLE	0.6124	0.6062	0.6175	0.3735	0.5063	0.5326
WANG ET AL.	0.6167	0.6207	0.6554	0.5062	0.5415	0.5982
PRIOR	0.5294	0.5081	0.5269	0.6707	0.6705	0.6838
SPEAGLE	**0.6905**	**0.6575**	**0.6710**	**0.7887**	**0.7695**	**0.7942**
SP'LE$^+$(1%)	0.7078	0.6828	0.6907	0.7951	0.7829	0.8040
SPLITE$^+$(1%)	0.6744	0.6542	0.6784	0.7693	0.7631	0.7923

the running times.

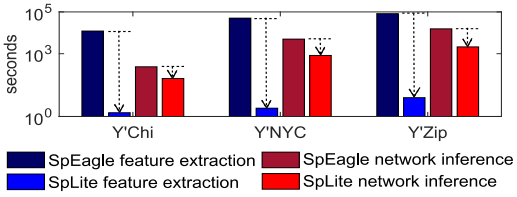


Figure 3: Runtime for SPEAGLE vs. SPLITE.

5. CONCLUSION

In this work, we propose a new holistic framework called SPEAGLE that exploits both relational data (review network) and metadata (behavioral and text) collectively to detect suspicious users, reviews, and targeted products. We evaluate our method on three real-world labeled (filtered vs. recommended) review datasets collected from Yelp.com. We provide the largest scale quantitative evaluation on opinion spam detection. Our results show that SPEAGLE is superior to several baselines and state-of-the-art techniques. We share our code and datasets with ground truth at `http://shebuti.com/collective-opinion-spam-detection/`.

6. REFERENCES

[1] L. Akoglu, R. Chandy, and C. Faloutsos. Opinion fraud detection in online reviews by network effects. In *ICWSM*, 2013.

[2] G. Fei, A. Mukherjee, B. Liu, M. Hsu, M. Castellanos, and R. Ghosh. Exploiting burstiness in reviews for review spammer detection. In *ICWSM*, 2013.

[3] S. Feng, R. Banerjee, and Y. Choi. Syntactic stylometry for deception detection. In *ACL*, 2012.

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

[5] F. Li, M. Huang, Y. Yang, and X. Zhu. Learning to identify review spam. In *IJCAI*, 2011.

[6] H. Li, Z. Chen, B. Liu, X. Wei, and J. Shao. Spotting fake reviews via collective positive-unlabeled learning. In *ICDM*, 2014.

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

[8] A. Mukherjee, A. Kumar, B. Liu, J. Wang, M. Hsu, M. Castellanos, and R. Ghosh. Spotting opinion spammers using behavioral footprints. In *KDD*. ACM, 2013.

[9] A. Mukherjee, V. Venkataraman, B. Liu, and N. S. Glance. What yelp fake review filter might be doing? In *ICWSM*, 2013.

[10] M. Ott, C. Cardie, and J. T. Hancock. Estimating the prevalence of deception in online review communities. In *WWW*, pages 201–210, 2012.

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

[12] G. Wang, S. Xie, B. Liu, and P. S. Yu. Review graph based online store review spammer detection. In *ICDM*, pages 1242–1247, 2011.

[13] J. S. Yedidia, W. T. Freeman, and Y. Weiss. Understanding BP and its generalizations. In *Exploring AI in the new millennium*. 2003.