Journal of Algorithms and Computation

journal homepage: http://jac.ut.ac.ir

Opinion Fraud Detection in Streaming Comments Utilising Node Similarity in the Review Network

Shahab Ghodsi*[1] and Ali Moeini†[2]

[1,2]University of Tehran, College of Engineering, School of Engineering Science, Tehran, Iran.

ABSTRACT

One important criterion in decision making when we want to purchase a product or a service is users' reviews. When something is valuable, it's fake will be created as well. It is the same for users' reviews. The purpose of these fake reviews is to deceive users, leading them to make a wrong choice. One challenging problem is when we can trust a review. Although many researchers attempted to address this problem, none of them pictured the problem in a streaming domain. With the help of the review network's properties, we propose a model to find reliable reviews when reviews are coming in a stream. Our model is fast and online, that is, it is capable of identifying reliable reviews as they are been submitted, and scalable because it is a complementary model to offline models in detecting fake reviews.

ARTICLE INFO

Article history:
Research paper
Received 18, September 2022
Received in revised form 09, November 2022
Accepted 05 December 2022
Available online 30, December 2022

Keyword: big data, opinion fraud, network science, review spam, Spark, fake reviews, streaming data.

AMS subject Classification: 05C78.

1 Introduction

Online reviews reflect users' opinions and experiences in using a product. Thus reviews can persuade users to buy a product or dissuade them [2, 21]. So having pleasant reviews is like a value-added for a product. Hence, there are enough reasons for fake reviews to be created. There are plenty of researches aiming to identify fraud reviews [8, 24, 11, 5, 3].

*shahab.ghodsi@ut.ac.ir

†Corresponding author: A. Moeinii. Email: moeini@ut.ac.ir

In these researches, the whole data must be available. However, there is a scenario when opinion fraud detection is needed as soon as the review is submitted. To be clear, suppose we have a snapshot of the reviews' network and the corresponding metadata and after, we receive streams of new reviews. The streams can append new users to the current network. Therefore, new reviews can be from existing users or new users to existing products. To the best of our knowledge, this scenario has not been addressed yet. We seek to build a model that is capable of designating reliable reviews in the upcoming streams. Thus, we need a measurement to distinguish the review's reliability. We design the model with the help of Spark, a multi-language engine for executing data engineering, data science, and machine learning on single-node machines or clusters. To evaluate our model, we compare the online results (when the data are streaming) with the offline results (when the entire data are available).

2 Background

Fabricating multiple fake reviews with different content is costly and time-consuming. This idea was first mentioned by Jindal and Liu [6]. Their work focused on finding duplicate reviews, they applied this technique to identify spam reviews. They manually labelled 470 reviews as fake according to their duplication score. Jindal and Liu [7] tried to improve their previous work. They Categorised spam reviews to three types: untruthful opinions, reviews on brands only and non-reviews. They crawled reviews from amazon.com and took advantage of Jaccard similarity for spam detection. Lai et al. (2010) tried to compare reviews and established a similarity score between them. The authors used SVM for spam detection. Many researchers took advantage of text features in review fraud detection. Ott et al. [19] tackled the problem using categorization and sentiment analysis of the text. They used supervised learning technique to detect opinion spam. they manually labelled reviews considering duplicate reviews as spam and the rest as genuine. One of the weaknesses of methods that focus on the textual information of reviews is the lack of generality. I.e. each domain (e.g. hotels, restaurants, applications, etc.) needs a new dataset and therefore, a new model. Lim et al. [14] presented a behavioural model to discover spammers. They proposed scoring methods to measure the degree of spam for each reviewer and apply them to an Amazon review dataset.
1 INTRODUCTION
Wang et al. [22] offered a novel concept of a heterogeneous review graph to capture the relationships among reviewers, reviews and stores. They developed an effective computation method to quantify the trustiness of reviewers, the honesty of reviews, and the reliability of stores. Xie et al. [23] proposed a time series pattern discovery mechanism for spam detection. Mukherjee et al. [15] observed behavioural footprints of reviewers, they developed an unsupervised model and called it Author Spamicity Model (ASM). Li et al. [13] suggested a Sparse Additive Generative (SAGE) Model based on generative Bayesian approach for detecting fake reviews. Akoglu et al. [1] proposed a framework for spotting fraudsters and fake reviews in online review datasets. Their idea was to use belief propagation to obtain beliefs for the nodes in the review network. Rayana and Akoglu [20] combined metadata information and the network review utilising the loopy belief Propagation approach. Li et al. [12] introduced a novel sentence weight neural network (SWNN) for assigning weights to user reviews' sentences. They used part of

speech tagging and person pronoun features in combination with SWNN. Dong et al. [4] presented an end-to-end trainable unified model to leverage the appealing properties from Autoencoder and random forest. They ran their model on Amazon review dataset. Tripathi et al. [10] provided a novel parallel bio-geography optimisation-based method to unfold the fake review detection in the big data environment. As shown in the literature, the common property of the recent works is offline processing of the data. We tend to generate an online algorithm for opinion fraud detection.

3 Proposed Method

In this section, we describe our model. The model comprises two parts, the offline part and the online part. First, we discuss the offline side algorithm then we explain the online side.

3.1 Problem Description

In brief, the problem is to assign reliability labels to streaming reviews. Before going any further, let us define review network. A review network is a graph like $G = (V, E)$, where $V = U \cup P$ denotes vertices set which is partitioned to users and products sets. $(u, p) \in E$ represents an edge, i.e., a review from user u for product p. Having the review network and the metadata (timestamps, products' categories, etc.) according to its nodes, we can make profiles for the nodes. A profile summarises a node's activities (e.g. average ratings of a node, maximum number of reviews in a day for a node, etc.). Thus, we can extract helpful features for fraud detection from the nodes' profiles and use them to detect fraudulent entities. This approach is only feasible when the whole data are available. In the case of streaming data, we cannot build the proper profiles since we don't have the entire data at once. Figure 1 pictures the problem; we have a snapshot of the network, and after that, five new reviews from two new users and two pre-existing ones. The green colour for users (products) indicates their honesty (goodness) level, whereas the red colour shows fraudulent (badness) level. The colour range of reviews identifies the reliability labels, which vary from red (Highly Not-Reliable) to green (Highly Reliable).

3.2 Offline Algorithm

On the offline side, the goal is to calculate a score that shows the spam degree of a user(product), when this score is large enough it means there is a high probability that the user (product) is a spammer (spammer's target). There are a lot of researches demonstrating different approaches to identify the spam score (). To keep it simple, we utilise an unsophisticated technique to compute sp. We use some behavioural features of users(products) [17, 16] and a couple of network's features. Behavioural features are the followings:

- Maximum Number of Reviews (MNR): Posting many reviews in a single day indicates an abnormal behaviour. Spammers write more reviews than normal users in a day, on the other hand, their targets get a large number of reviews too.
- Percentage of Positive Reviews (PR): Usually, spammers try to hype a product that causes most of their reviews to be positive.

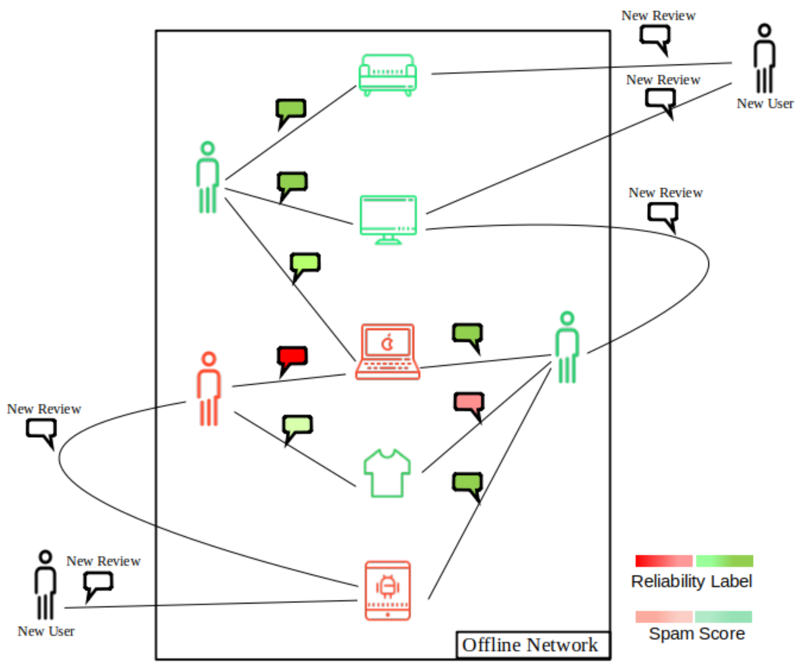

Figure 1: Offline data: A review network with 2 honest users, 3 good products, 1 fraudulent user and 2 bad products. Online data: 3 reviews from 2 new users and 2 reviews from 2 pre-existing users.

- Percentage of Negative Reviews (NR): Some spammers focus on defaming a product, so most of their reviews are negative.
- Average Rating Deviation (avgRD): Review spamming usually involves wrong projection either in the positive or negative light so as to alter the true sentiment on products. This hints that ratings of spammers often deviate from the average ratings given by other reviewers. Fraudster users and fake products have a high value of avgRD.

To extract the network's features, we run the hubs and authorities algorithm on our graph [9]. Consequently, we get two scores for a node: authority score and hub score: authority score(aScore) and hub score(hScore). In our problem when the aScore of a node is high, it means the node is an important product in the network and if the hScore is high, it implies that the node is an influential user. It is trivial that spammers and their targets have lower value of the hScore and the aScore respectively.
Having introduced the features, the next step is to provide a probability value for each of them and then calculate the s_p based on the probabilities. We split the features into two sets, $F_H = \{MNR, PR, NR, avgRD\}$ and $F_L = \{aScore, hScore\}$. A spam node is more likely to have higher values of F_H features and lower values of F_L features. Expressions 1 and 2 show the probability functions.

$$\forall i \in V, \forall j \in F_H; P_{ij} = \begin{cases} \dfrac{f_{ij}}{max(j)} & f_{ij} \geq avg(j) \\ 0 & otherwise \end{cases} \tag{1}$$

$$\forall i \in V, \forall j \in F_L; P_{ij} = \begin{cases} 1 - f_{ij} & f_{ij} \leq avg(j) \\ 0 & otherwise \end{cases} \tag{2}$$

Note that aggregate functions, $max()$ and $avg()$, work based on the type of the node. I.e. their return values depend on the node's type. E.g., if i is a user node, $max(j)$ finds maximum value of j over all the user nodes. Expression 3 defines spam score of the nodes.w_j represents weight of the probability, this value estimates the importance of the feature. Because our focus is on the online part of the algorithm, we don't dive into computing these weights so we initialise all of them to 1.

$$\forall i \in V, sp_i = \frac{\sum_j w_j P_{ij}}{\sum_j w_j} \tag{3}$$

3.3 Online Algorithm

At this point, we have all the necessary information for the online algorithm. In this algorithm, we aim to choose an appropriate label showing the reliability of the reviews when they just get submitted. As previously noted, the online nature of our problem creates two cases: the first is when an existing user in the current network writes a review and the latter is a new user that is not in the current network doing so. The first case is straightforward because we know the profile of the user and we have summarised his behaviour in the , so there is no trouble in decision making. but the second case requires some extra steps. Since there is no piece of information about the user, in this case, we don't have any prior knowledge about him. We use the user similarity to solve the issue. For similarity measurement, we determine Euclidean distance between the new user and all the present users that have submitted reviews for the same product. Dimensions of this distance are: rating for the product, the time when the review is submitted, verification status of the review, i.e., whether the user has purchased the product or not and the degree of the user in the graph (for new users, the degree is 1). For each dimension, we consider a coefficient. The coefficient designates the dimension importance in the distance evaluation. For example, the rating dimension has a higher priority than the degree dimension. Thus we initiate the coefficients of more important dimensions to 2 and the rest to 1. The final step of the algorithm is decision making. We estimate the reliability label based on three factors: user's *sp* value, product's *sp* value and the rating deviation from average rating of the product (see Algorithm 2).

4 Dataset

Amazon review dataset was first released in 2014. It includes reviews (ratings, text, helpfulness votes), product metadata (descriptions, category information, price, brand, and image features), and links (also viewed/also bought graphs) [18]. It consists of several categories like book, electronic, software, etc. For our purpose, we choose video games category and run our algorithm on it. Table 1 shows some statistics about this category.

Algorithm 1: Online opinion fraud detection

```
def evaluateReview:
    Data: userX, productX
    Result: The reliability label
    if userX is a new user then
        similarUser = findTheMostSimilarUser(userX, productX)
        return isReliable(similarUser, productX)
    else
        return isReliable(userX, productX)

def findTheMostSimilarUser:
    Data: userX, productX
    Result: The most similar user to userX
    foreach user in the productX neighbourhood do
        calculate distance(user, userX)
    return the user with the minimum distance(user, userX)

def distance:
    Data: userA, userB
    Result: distance between userA and userB
    // c1 ... c4 are the influence coefficients of the distances
    return $sqrt(c1(rating\_distance)^2 + c2(datetime\_distance)^2 +$
      $c3(nodeDegree\_distance)^2 + c4(isVerified\_distance)^2)$

def isReliable:
    Data: userX, productX
    Result: The reliability label
    $deviate$ = (abs(userX.R - productX.avgR) > $avg_{products}(avgRD)$)
    if $userX.sp > 0.5$ then
        if deviate then
            return Highly Not-Reliable
        return Not-Reliable
    if $0.3 \leq userX.sp \leq 0.5$ then
        if deviate then
            if $productX.sp > 0.5$ then
                return Not-Reliable
            return Fairly Not-Reliable
        return Reliable
    if $userX.sp < 0.3$ then
        if deviate then
            return Fairly Reliable
        return Highly Reliable
```

Algorithm 2: Online opinion fraud detection

```
def evaluateReview:
    Data: userX, productX
    Result: The reliability label
    if userX is a new user then
        similarUser = $findTheMostSimilarUser$(userX, productX)
        return isReliable(similarUser, productX)
    else
        return isReliable(userX, productX)

def findTheMostSimilarUser:
    Data: userX, productX
    Result: The most similar user to userX
    foreach user in the productX neighbourhood do
        calculate distance(user, userX)
    return the user with the minimum distance(user, userX)

def distance:
    Data: userA, userB
    Result: distance between userA and userB
    // c1 ...  c4 are the influence coefficients of the distances
    return $sqrt(c1(rating\_distance)^2 + c2(datetime\_distance)^2 +$
      $c3(nodeDegree\_distance)^2 + c4(isVerified\_distance)^2)$

def isReliable:
    Data: userX, productX
    Result: The reliability label
    deviate = (abs(userX.R - productX.avgR) > $avg_{products}(avgRD)$)
    if $userX.sp > 0.5$ then
        if deviate then
            return Highly Not-Reliable
        return Not-Reliable
    if $0.3 \leq userX.sp \leq 0.5$ then
        if deviate then
            if $productX.sp > 0.5$ then
                return Not-Reliable
            return Fairly Not-Reliable
        return Reliable
    if $userX.sp < 0.3$ then
        if deviate then
            return Fairly Reliable
        return Highly Reliable
```

Table 1: Amazon Video games dataset statistics

Component	#number	Max degree	Average rating
Users	1′540′618	841	
Products	71′982	6′462	
Reviews	2′490′986		4.02

5 Results

In this section, we explain similarity calculation in more detail and show the importance of the coefficients with an example. Then, we describe our strategy to simulate streams of reviews. After, we present the spam scores of the nodes in the offline review network and the online reliability labels. We proceed by evaluating the online results against the offline. In the end, we demonstrate some reviews along with their reliability labels. We have discussed the similarity in the previous section. Assume we have the review network shown in Table 2; we want to find the most similar user to user U4 who has just submitted his review for product p1 (Figure 1). Since the reviewed product is P1, we have to search for the most similar user in the neighbourhood of P1. So, our candidate users for similarity calculations are U1, U2 and U3. Which user do you choose if you were asked to find the most similar user to U4? Probably your answer would be U3 because the first idea that comes to your mind is to compare users' ratings. Hence, some fields in comparing two users have higher importance. These fields in our case are: "rating" and "verified". The importance of the first one is obvious. The "verified" parameter is influential because we want to use the spam score of the most similar user instead of the new user, so these two users should be almost identical in the "verified" parameter. A user whose purchase is verified is less likely to be fraudulent. To emphasise the fields' importance in the similarity, we initialise c1 and c4 to 2 and c2 and c3 to 1. This action means that the "rating" and the "verified" fields have higher priorities than the other two fields in the comparison. Table 3 shows the distances between U4 and the candidates with the coefficients' effect and without it. Applying the coefficients on the dimensions, returns a more accurate and meaningful result.

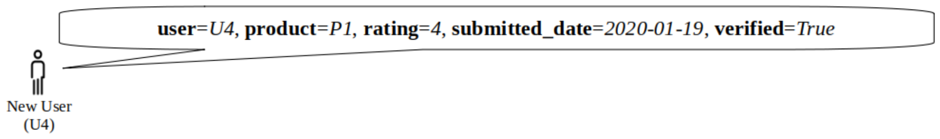
user=U4, product=P1, rating=4, submitted_date=2020-01-19, verified=True
New User (U4)

Figure 2: A new user's opinion; user U4 has just submitted his opinion

We divide our data into two parts: (A) a part for the offline process (80% of the data) and (B) a part to simulate streams of reviews (20% of the data). It is worth noticing that we use uniform sampling for the partitioning, i.e. each review has an equal chance to get picked in the streaming part. First, we create the review network from (A) and determine the spam scores for the users and the products in the network (offline algorithm). With the help of Spark, we produce our streams from (B). A stream is a random number of

Table 2: A review network with 3 users, 3 products and 6 reviews

src	dst	rating	Submitted_date	user_degree	verified
U1	P1	5	2020-03-23	2	False
U1	P2	4	2020-03-29	2	True
U2	P1	2	2022-01-07	1	True
U3	P1	4	2021-10-18	3	True
U3	P2	1	2021-08-25	3	False
U3	P3	2	2021-11-13	3	True

Table 3: Distance Calculation for 3 candidates with and without coefficient effect; distances with coefficient effect are more accurate and reflect a more sensible meaning

user	candidate	distance	
		Without coefficient effect	With coefficient effect
U4	U1	$\sqrt{(4-5)^2+(2022-2020)^2+(1-2)^2+(1-0)^2}=\sqrt{7}$	$\sqrt{2*(4-5)^2+1*(2022-2020)^2+1*(1-2)^2+2*(1-0)^2}=\sqrt{9}$
U4	U2	$\sqrt{(4-2)^2+(2022-2022)^2+(1-1)^2+(1-1)^2}=\sqrt{4}$	$\sqrt{2*(4-2)^2+1*(2022-2022)^2+1*(1-1)^2+2*(1-1)^2}=\sqrt{8}$
U4	U3	$\sqrt{(4-1)^2+(2022-2021)^2+(1-3)^2+(1-1)^2}=\sqrt{5}$	$\sqrt{2*(4-1)^2+1*(2022-2021)^2+1*(1-3)^2+2*(1-1)^2}=\sqrt{5}$

reviews varying from 1 to 100, at a second. At each stream, we save the reviews along with the estimated labels.

Figure 3 shows the degree distribution of the nodes in the online and the offline parts. As you can see, there is a skewness akin to the skewness of power-law in both parts. Figure 3 (b) indicates that approximately 50% of the users have a degree of 1. As previously mentioned, the degree of new users is 1, so these users are considered new users in our work.

When the streams ended, we run the offline algorithm on all the data (100% of the data) in order to get the spam scores for the nodes in (B). Then, we compute reliability labels based on their real spam scores (unlike the online algorithm that labels are calculated based on the most similar user's spam score). Eventually, we compare the streaming (online) results with the offline ones (see Table 4). 97% of the reviews get the identical label in online and offline results. We find out 46% of the different labels, actually have the same direction but different levels of intensity. E.g., Not-Reliable and Fairly Not-Reliable both indicate that the review is not reliable but they differ in the amount of unreliability.

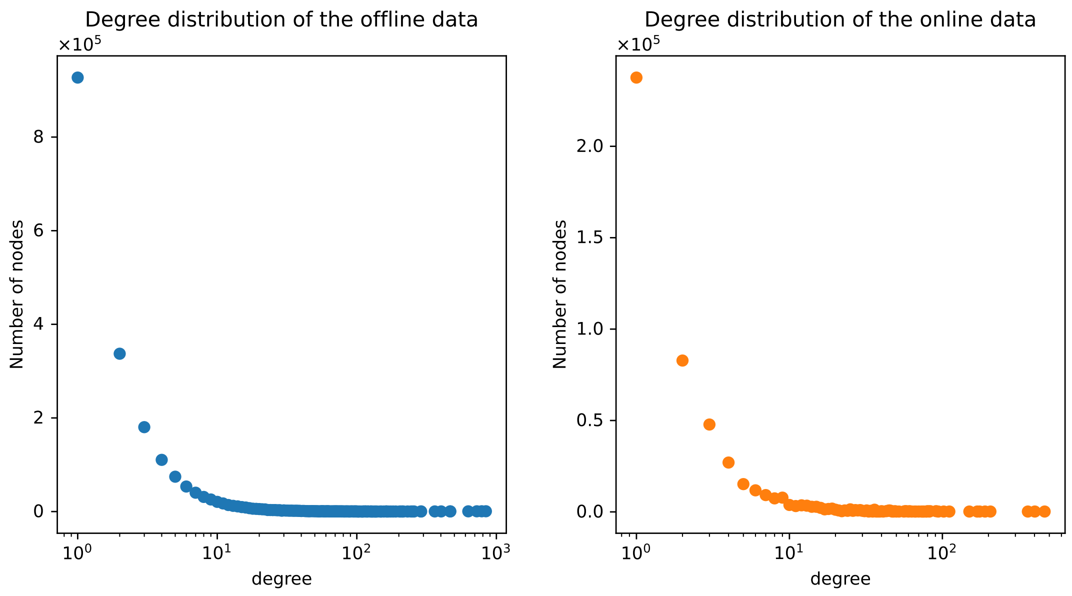

Figure 3: Degree distribution in (a) the offline data on the left plot and (b) the online data on the right plot

Table 4: label's comparison of the same piece of data when it is online against when it is offline

Exactly The same label	Not the same label	
97%	3%	
	The Same direction	Not the same direction
	46%	54%

Figure 4 illustrates the spam distribution of the spam scores. We observe that the majority of nodes are almost neutral (i.e. their spam score is equal to 0.5). Since a high percentage of reviews are from singleton users (i.e. users with only 1 review in the review network), we can perceive a more considerable number of the products has a spam score less than or equal to 0.2 than users (compare (a) and (b) in Figure 4).

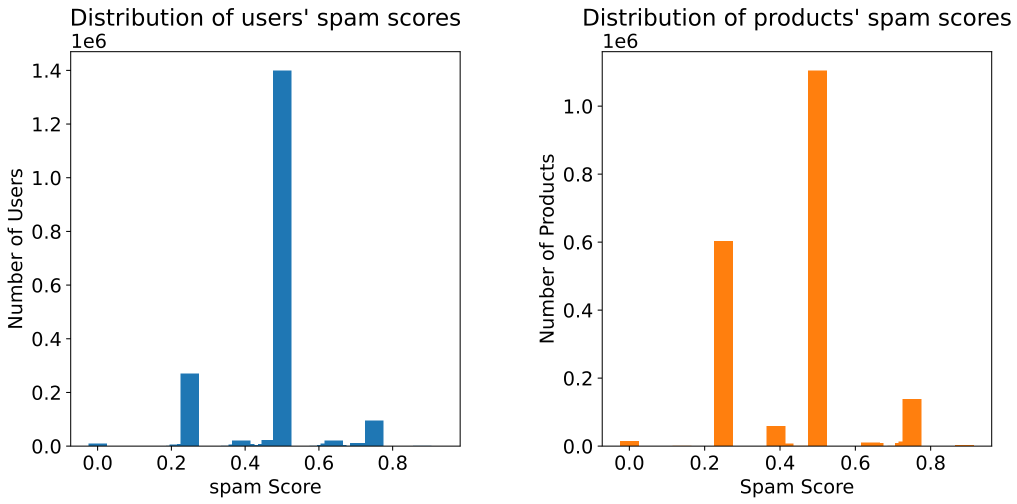

Figure 4: The spam score distribution in (a) the users on the left plot and (b) the products on the right plot

Figure 5 displays the distribution of the reliability labels in the streams of reviews. It makes sense that only around 3% of the reviews are highly reliable.

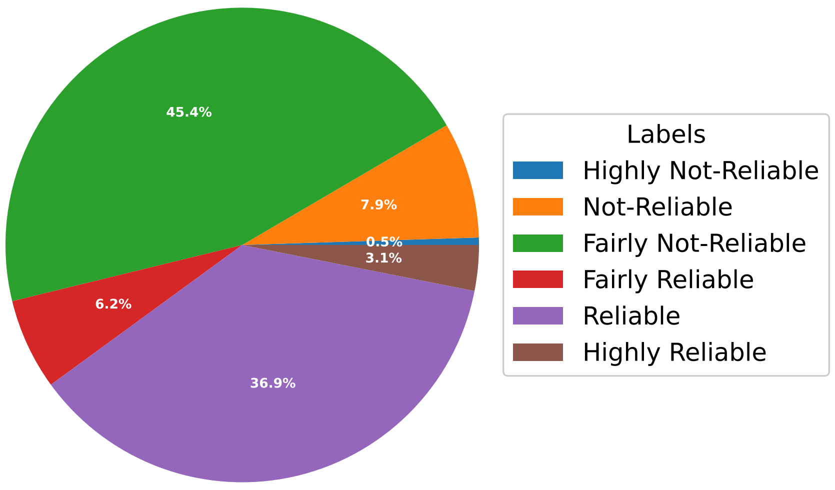

Figure 5: Reliability labels' distribution of the streams

We scrutinise the different labels in online and offline results in Figure 6. We discover 76% of the labels that differ in the reliability direction, flip between Not-Reliable and Fairly Reliable.
Finally, Table 5 shows a sample of reliability labels in the streaming data together with the actual video games' reviews in the Amazon review dataset.

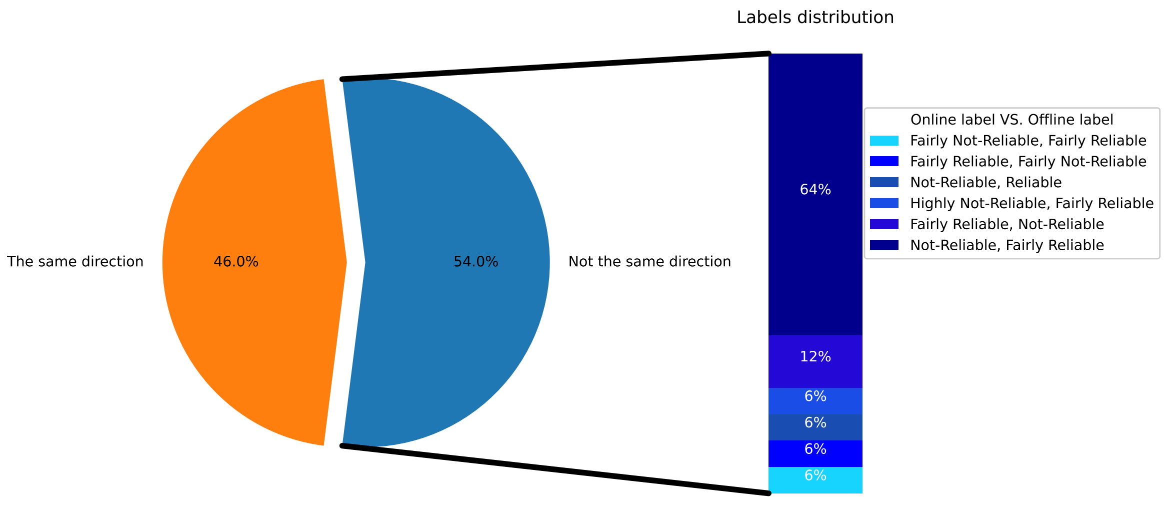

Figure 6: Distribution of the reliability labels that vary from the offline labels

Table 5: label's comparison of the same piece of data when it is online against when it is offline

Label	Rating	Text
Fairly Not-Reliable	5	Happy little boy
Not-Reliable	5	Thanks
Highly Not-Reliable	1	break easily
Fairly Reliable	5	This works. It does what it is supposed to and I never have to worry about batteries for my 360. yay!
Reliable	2	The graphics on the game are good, and as a basic poker game it passes, but if you're looking to practice your skills with this game, it disappoints. All of the computer players stay in with basically any hand, regardless of the level of play. I've been beaten by too many Two-Eight off suits to mention.
Highly Reliable	4	My 5 year old daughter LOVES this game. It is really cute and easy to understand. It has many levels and activities for the player to explore. I recommend it!

6 Conclusion

Having a real-time spammer detection is a bless. But it is costly and hard to maintain. The effectiveness of the model highly depends on the offline model you choose to use as the base for the online model. So the more accurate the offline model is, the more powerful the online model gets. On the other hand, an online algorithm for reliability checking of the reviews should be tremendously accurate, because it affects the users' decisions.

References

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

[2] Aslam, U., Jayabalan, M., Aziz, H. I., and Sohail, A. A survey on opinion spam detection methods. *International Journal of Scientific & Technology Research 8* (09 2019), 1355–1363.

[3] Dhingra, K., and Yadav, S. K. Spam analysis of big reviews dataset using fuzzy ranking evaluation algorithm and hadoop. *International Journal of Machine Learning and Cybernetics* (2019).

[4] Dong, M., Yao, L., Wang, X., Benatallah, B., Huang, C., and Ning, X. Opinion fraud detection via neural autoencoder decision forest. *Pattern Recognition Letters 132* (2020), 21–29. Multiple-Task Learning for Big Data (MTL4BD).

[5] Guo, Z., Tang, L., Guo, T., Yu, K., Alazab, M., and Shalaginov, A. Deep graph neural network-based spammer detection under the perspective of heterogeneous cyberspace. *Future Generation Computer Systems 117* (04 2021), 205–218.

[6] Jindal, N., and Liu, B. Analyzing and detecting review spam. In *Seventh IEEE International Conference on Data Mining (ICDM 2007)* (2007), pp. 547–552.

[7] Jindal, N., and Liu, B. Opinion spam and analysis. In *Proceedings of the 2008 International Conference on Web Search and Data Mining* (New York, NY, USA, 2008), WSDM '08, Association for Computing Machinery, p. 219–230.

[8] Kauffmann, E., Peral, J., Gil, D., Ferrández, A., Sellers, R., and Mora, H. A framework for big data analytics in commercial social networks: A case study on sentiment analysis and fake review detection for marketing decision-making. *Industrial Marketing Management 90* (2020), 523–537.

[9] Kleinberg, J. M. Authoritative sources in a hyperlinked environment. *J. ACM 46*, 5 (sep 1999), 604–632.

[10] Kumar Tripathi, A., Sharma, K., and Bala, M. Fake review detection in big data using parallel bbo. *International Journal of Information Systems & Management Science 2*, 2 (2019).

[11] Li, H., Chen, Z., Mukherjee, A., Liu, B., and Shao, J. Analyzing and detecting opinion spam on a large-scale dataset via temporal and spatial patterns.

[12] Li, H., Fei, G., Wang, S., Liu, B., Shao, W., Mukherjee, A., and Shao, J. Bimodal distribution and co-bursting in review spam detection. In *Proceedings of the 26th International Conference on World Wide Web* (Republic and Canton of Geneva, CHE, 2017), WWW '17, International World Wide Web Conferences Steering Committee, p. 1063–1072.

[13] Li, J., Ott, M., Cardie, C., and Hovy, E. Towards a general rule for identifying deceptive opinion spam. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)* (Baltimore, Maryland, June 2014), Association for Computational Linguistics, pp. 1566–1576.

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

[15] Mukherjee, A., Kumar, A., Liu, B., Wang, J., Hsu, M., Castellanos, M., and Ghosh, R. Spotting opinion spammers using behavioral footprints. In *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (New York, NY, USA, 2013), KDD '13, Association for Computing Machinery, p. 632–640.

[16] Mukherjee, A., Liu, B., and Glance, N. Spotting fake reviewer groups in consumer reviews. In *Proceedings of the 21st International Conference on World Wide Web* (New York, NY, USA, 2012), WWW '12, Association for Computing Machinery, p. 191–200.

[17] Mukherjee, A., Venkataraman, V., Liu, B., and Glance, N. What yelp fake review filter might be doing? *Proceedings of the International AAAI Conference on Web and Social Media 7*, 1 (2013), 409–418.

[18] Ni, J., Li, J., and McAuley, J. Justifying recommendations using distantly-labeled reviews and fine-grained aspects. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)* (Hong Kong, China, Nov. 2019), Association for Computational Linguistics, pp. 188–197.

[19] Ott, M., Choi, Y., Cardie, C., and Hancock, J. T. 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* (Portland, Oregon, USA, June 2011), Association for Computational Linguistics, pp. 309–319.

[20] Rayana, S., and Akoglu, L. Collective opinion spam detection: Bridging review networks and metadata. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (New York, NY, USA, 2015), KDD '15, Association for Computing Machinery, p. 985–994.

[21] Ren, Y., and Ji, D. Learning to detect deceptive opinion spam: A survey. *IEEE Access PP* (04 2019), 1–1.

[22] Wang, G., Xie, S., Liu, B., and Yu, P. S. Identify online store review spammers via social review graph. *ACM Trans. Intell. Syst. Technol. 3*, 4 (sep 2012).

[23] Xie, S., Wang, G., Lin, S., and Yu, P. S. Review spam detection via temporal pattern discovery. In *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (New York, NY, USA, 2012), KDD '12, Association for Computing Machinery, p. 823–831.

[24] Xu, C., Zhang, J., and Sun, Z. Online reputation fraud campaign detection in user ratings. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence* (2017), IJCAI'17, AAAI Press, p. 3873–3879.