HawkEye: A Robust Reputation System for Community-based Misinformation Detection

Rohit Mujumdar, Srijan Kumar
Georgia Institute of Technology
{rohitmujumdar,srijan}@gatech.edu

***Abstract*—Identifying misinformation is a critical task on web and social media platforms. Recent efforts have focused on leveraging the community of ordinary users to detect, counter, and curb misinformation. Twitter has recently launched a community-driven misinformation detection service called Birdwatch. On Birdwatch, users can provide notes to label tweets as misleading or not, and they can rate other users' notes as being 'helpful' or not. However, malicious users can inject fake notes and helpfulness ratings to manipulate the system for their gains. In this work, we investigate the robustness of Birdwatch against adversaries. Using the entire datasets of Birdwatch, we show that the current Birdwatch system is vulnerable to adversarial attacks — using only a few fake accounts, an adversary can promote any random note as one of the top ranking notes. To overcome this vulnerability, we develop HawkEye. We propose four metrics to determine the intrinsic quality of users, notes, and tweets. We propose seven axioms that these quality scores should follow. We propose a cold-start-aware graph-based recursive algorithm, which satisfies the axioms, to measure all the quality scores. We show that HawkEye is more robust against adversarial manipulation. We also show that HawkEye outperforms Birdwatch in identifying accurate and misleading tweets.**

INTRODUCTION

Misinformation is rampant on web and social media platforms as users can create and spread fake news and misleading content [1, 2, 3, 4]. Such content can cause severe harm to the democracy and society [3], disrupt belief in public institutions [5], cause mass panic [6], and threaten public health [6, 7]. In some cases, they lead to real-world violence [8]. As misinformation spreads wide and fast [2], it becomes crucial to curb and prevent its spread.

Several machine learning techniques have been created to detect misinformation and disinformation [1, 9, 10]. These methods use the text of the misinformation content, propagation patterns, temporal information, and other features. However, misinformation can be written to escape detection from these automated techniques [10].

On the other hand, humans can identify misinformation when they see it, even if a machine learning algorithm can not [11, 12, 13]. Several recent studies have shown how users can help to identify potential misinformation [11, 13, 14]. Users can act as 'eyes on the ground' and can help counter-misinformation efforts in a variety of different ways — they can flag or report it, they can reply to the misinformation message with corrective information, or they can engage with the users exposed to the misinformation to provide them true information. This power of the community of ordinary users can be harnessed to identify, remove, and limit the spread of misleading content on social media.

Along these lines, Twitter has launched a new community-driven service to address misinformation called Birdwatch [1]. Twitter users can enroll on Birdwatch to rate tweets according to their accuracy. Users can identify the tweets they believe are misleading (or not misleading), write notes that provide context for their categorization, and even rate the helpfulness of other users' notes. Notes are ranked according to their quality, as determined by the helpfulness ratings given by other users. Tweets can be labeled as misinformative if there are enough number of high-helpfulness notes that say so. However, with any system that depends on user inputs, the Birdwatch system can be vulnerable to malicious actors—such actors can add fake ratings to notes to increase or decrease its perceived quality, and consequently the classification of the tweet. It is thus essential to understand the vulnerabilities of the Birdwatch system against adversaries.

Present work. Here we study the vulnerability of the community-based misinformation detection platform, Birdwatch (by Twitter). We formulate the community feedback ecosystem as a multipartite graph, containing user nodes, note nodes, tweet nodes, and rating edges. We propose intrinsic quality metrics for different entities in the graph, namely accuracy of tweets, credibility of notes that are written for tweets, and the trustworthiness of the users who write or rate notes. We propose a set of seven axioms that these quality metrics should satisfy and propose a formulation that satisfies these axioms. We propose HawkEye, a graph-based recursive algorithm that leverages the global graph structure to quantify all the quality metrics. Since many users will only write and rate a few notes and many tweets will only have a few notes, we introduce a Laplacian smoothing technique to overcome this cold-start problem. We posit that HawkEye will be more robust to adversaries.

We answer three research questions:

- **Research Question 1:** Is Birdwatch vulnerable to adversarial attacks that manipulate note rankings?
- **Research Question 2:** Is the proposed HawkEye algorithm more robust to attacks?
- **Research Question 3:** How accurate are the HawkEye and Birdwatch algorithms in identifying misleading tweets?

[1] https://birdwatch.twitter.com/

We compare the Birdwatch and HawkEye models' robustness against an attacker whose goal is to manipulate the ranking of notes. We use Birdwatch's entire dataset to conduct experiments. Through extensive evaluation, we show that the existing reputation algorithm that Birdwatch uses is susceptible to attacks. An attacker can promote any random note's ranking by injecting a few fake ratings. Next, we show that our proposed HawkEye algorithm is more robust against this attack. Furthermore, we show that the HawkEye algorithm performs better than the Birdwatch system in identifying accurate and misleading tweets in both unsupervised and supervised settings.

The code and data used in the experiments will be made publicly available upon publication.

Related work

In this section, we introduce the relevant related works.

Community-based misinformation detection. A community's role in combating misinformation online as large-scale fact-checkers has been studied recently [15, 16, 17]. Others describe how crowd-sourced expertise can act as weak supervision for automated misinformation detection systems [18, 19] or in augmenting machine learning driven fact-checking [20, 21]. Other works [13, 22] provide insight into how misinformation can organically be countered by the crowd by amplifying the voices of professional fact-checkers, who often have limited reach.

Graph-based reputation systems. Graphs have been used to build reputation systems to detect malicious users and unwanted behavior, such as fake reviewers [23, 24, 25], spammers [26], auction fraud [27], and retweet fraud [28]. While fraudulent behavior has been studied in crowdsourcing systems [29, 30], few of them use graphs to build reputation systems for crowdsourced systems. Additionally, no graph-based reputation system exists for the Birdwatch system by Twitter.

Birdwatch Description and Dataset

We describe how Birdwatch works and then its data.[2]

The Birdwatch system relies on users contributing in two ways: notes and ratings. A note is a label given by a user to a tweet, and a rating is a label given by a user to a note. A Birdwatch user writes notes for tweets in their timeline, either to label a tweet spreading misinformation as 'misleading' or a tweet not spreading any misinformation as 'not misleading'. Notes can be associated with information where participants can optionally explain why they believe a tweet is misleading, and provide links to relevant sources. We observed that very few users provide such additional information. For the purpose of our experiments, we focus only on the note's binary verdict (misleading or not-misleading) provided to the tweet.

Since users can give wrong notes (i.e. write notes marking a misleading tweet as not-misleading and vice-versa), Birdwatch also allows users to rate the helpfulness of notes written by other users. Users can rate a note as either 'helpful' or 'not helpful'. A user can not rate its own note. These ratings help identify which notes are the most helpful, as judged by other users, and allow Birdwatch to raise the visibility of those found to be the most helpful by many users. A *Note Helpfulness Ratio* is computed for each note as the proportion of ratings that say the note is helpful. Birdwatch develops a reputation system such that the notes which have at least 5 ratings and a note helpfulness ratio of at least 0.84 are deemed to be 'Currently Rated Helpful' (CRH). Among these, the top 5 notes, ranked according to their helpfulness ratio, are deemed to be the most credible and are most prominently displayed on the Birdwatch site next to tweets.

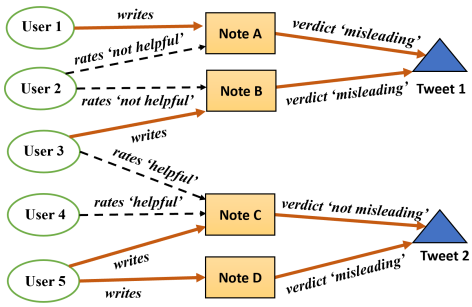


Fig. 1: We model the Birdwatch network as a directed multi-partite graph.

Dataset. We downloaded all the notes and ratings data made publicly available by Birdwatch. The data used for our experiments ranges from January 23, 2021 (when this feature was introduced) to April 13, 2021. This dataset contains a total number of 6,271 Birdwatch notes, given by 1472 users, to 4,900 tweets. A total of 17,682 ratings were given to notes by 1,368 users. A total of 1,895 users are present in the dataset. Each of the 4,900 tweets has at least one note written, and 3,445 notes have at least one rating.

Multipartite Graph. We model the Birdwatch data as a directed multipartite graph between users, notes, and tweets as shown in Figure 1. We define the Birdwatch graph $G = \{\mathcal{U}, \mathcal{N}, \mathcal{R}, \mathcal{T}\}$ as a directed multi-partite graph where $\{\mathcal{U}, \mathcal{N}, \mathcal{R}, \mathcal{T}\}$ represent the set of all users, notes, ratings and tweets, respectively. A user $u \in \mathcal{U}$ writes note $n \in \mathcal{N}$ or rates note $n \in \mathcal{N}$. Every note $n \in \mathcal{N}$ is written for a tweet $t \in \mathcal{T}$, and every rating $r \in \mathcal{R}$ is given for a note. There may be notes with no ratings. All tweets in the dataset have at least one note. A users can either write notes, rate notes, or do both.

We define $helpfulness(u, n) \in \{-1, 1\} \forall (u, n) \in \mathcal{R}$ as the helpfulness rating given by user u to note n: a score of 1 denotes the user rated the note as helpful, and -1 (not helpful) otherwise. Let $Out_r(u)$ be the set of all the ratings given by the user u and $In(n)$ be the set of all the ratings received by note n. Then, $|Out_r(u)|$ and $|In(n)|$ represent their respective counts. We further define $In^+(n)$ as all the ratings received by note n which are 'helpful' and $In^-(n)$ as all the ratings received by note n which are 'not helpful'.

We further define $verdict(n, t) \in \{-1, 1\}$ as the note n's

[2] As Birdwatch is an evolving system, some of its aspects may change over time. The description we provide below is the state of the Birdwatch system as of June 10, 2021.

classification of the 'misleadingness' of the tweet t. A score of -1 (negative verdict) denotes that the user giving the note believes the tweet is potentially misleading or spreading false information. A score of 1 (positive verdict) denotes that the tweet is not misleading. Let $In(t)$ refer to the set of all notes received by tweet t and $Out_n(u)$ be the set of all notes written by user u . Then, $|In(t)|$ and $|Out_n(u)|$ represent their respective counts. We further define $In^+(t)$ as all the 'not misleading' notes received by tweet t and $In^-(t)$ as all the 'misleading' notes received by tweet t.

Definition 1. [Identical Egonetworks for notes] : Two notes n_1 (given to tweet t_1) and n_2 (given to tweet t_2) have identical egonetworks if $|In(n_1)| = |In(n_2)|$ and there exists a one-to-one mapping $h : In(n_1) \rightarrow In(n_2)$ such that $helpfulness(u, n_1) = helpfulness(h(u), n_2) \forall (u, n_1) \in In(n_1)$ and $verdict(n_1, t_1) = verdict(n_2, t_2)$.

Definition 2. [Identical Egonetworks for tweets] : Two tweets t_1 and t_2 have identical egonetworks if $|In(t_1)| = |In(t_2)|$ and there exists a one-to-one mapping $h : In(t_1) \rightarrow In(t_2)$ such that $verdict(n, t_1) = verdict(h(n), t_2) \forall (n, t_1) \in In(t_1)$.

HawkEye Metric Formulation

We propose that users, notes, and tweets have (unknown) intrinsic scores that quantify their quality: user trust, note credibility, and tweet accuracy. In doing so, we take inspiration from relevant prior works [23, 25]. These scores are unknown apriori, but essential to formulate and quantify. We formally define these quality metrics below.

• **User Trustworthiness in Writing Notes**: Users vary in terms of how trustworthy they are when they are writing a note. Trustworthy users should ideally write a 'misleading' note to a ground-truth misinformation tweet and a 'non-misleading' note to a ground-truth accurate tweet. However, untrustworthy users can aim to do the opposite, with the goal of misguiding the audience — their note can lead to an accurate tweet being flagged for deletion, while keeping an inaccurate tweet in circulation. We define a metric called 'User Trustworthiness in Writing' a note as $TW(u)$ to quantify the expected quality of notes written by a user u. This scalar score lies between -1 and +1 (both inclusive) — a score of 1 denotes a user who always writes credible notes, while -1 means the opposite.

• **User Trustworthiness in Rating Notes**: A user who is trustworthy in writing notes may not necessarily be trustworthy when it comes to rating notes. For example, malicious users may write credible notes but may manipulate the ratings of note written by others — they may assign high helpfulness ratings to low quality notes, so that others may believe that the notes are high quality. Simultaneously, they may give ground-truth high quality notes a low rating.

Hence, we need a separate trustworthiness score for users in rating notes. We posit that users vary in terms of how trustworthy they are when they rate the quality of notes written by other users. We define a metric called 'Trustworthiness in Rating' $TR(u)$ to quantify a user u's expected trust in giving accurate ratings. This value lies between 0 and 1 (both inclusive), where higher values indicate higher trustworthiness.

• **Note Credibility**: As mentioned previously, in order to manipulate rankings, users can write low (high) scoring notes for high (low) quality tweets. While a user's writing trustworthiness measures the average quality of the user across all the notes she has written, the purpose of the Note Credibility score is to measure the quality of each individual note. The need to tease apart user's trustworthiness in writing from note's credibility arises from the fact that even a generally trustworthy (untrustworthy) user may sometimes write low (high) credibility note. Clearly, the same metric can not be used to quantify the quality of both users and notes. Therefore, the note credibility score $NC(n)$ is a score that lies between -1 and +1 (both inclusive).

• **Tweet Accuracy** : The Birdwatch system has been primarily built to identify tweets which could be misleading. Naturally, not all tweets are accurate. We define a metric called 'accuracy' to measure how correct its content is. The accuracy of a tweet is a number lying between -1 and +1 (both inclusive), where -1 denotes a tweet of the lowest accuracy and +1 denotes a note of the highest accuracy. Note that while the ground-truth of a tweet is either -1 (false tweet) or +1 (true tweet), our metric gives a score in the entire range of -1 to +1 to quantify the model's confidence in its label.

Having established the intrinsic quality scores of all the entities with ego networks, we can create qualitative definitions of ego networks.

Definition 3. [Identically Credible Egonetworks for tweets]: We say that two tweets t_1 and t_2 have identically credible egonetworks if $|In(t_1)| = |In(t_2)|$ and there exists a one-to-one mapping $h : In(t_1) \rightarrow In(t_2)$ such that $NC(n_1) = NC(h(n_1)) \forall (n_1, t) \in In(t_1)$.

Definition 4. [Identically Trustworthy Egonetworks for notes]: We say that two notes n_1 (written by u_1) and n_2 (written by u_2) have identically trustworthy writing egonetworks if $TW(u_1) = TW(u_2)$. Similarly, two notes n_1 and n_2 have identically trustworthy rating egonetworks if $|In(n_1)| = |In(n_2)|$ and there exists a one-to-one mapping $h : In(n_1) \rightarrow In(n_2)$ such that $TR(u) = TR(h(u)) \forall (u, n_1) \in In(n_1)$.

Since users, notes, and tweets are interdependent on one another, their quality scores are also recursively interdependent. We propose the following axioms that establish this interdependency and also lay ground for the metrics that we would be formulating for each of the quality scores. These axioms assume that all users are benign.

Axiom 1 (More accurate tweets get positive verdict notes): If two tweets have identically credible egonetworks and for the first tweet, more notes have positive verdicts ('not misleading') than the second tweet, then the first tweet is more accurate. Formally, if two tweets have a one-to-one mapping $h : In(t_1) \rightarrow In(t_2)$ such that $NC(n) = NC(h(n))$ and $verdict(n, t_1) >= verdict(h(n), t_2) \forall (n, t_1) \in In(t_1)$ then $A(t_1) >= A(t_2)$.

The second axiom defines the relation between a tweet and the credibility of the notes written for it.

Axiom 2 (More accurate tweets get higher credibility notes with the verdict 'not-misleading'): If two tweets have identical egonetworks and for the first tweet, the positive verdict ('not misleading') notes have higher credibility than the second tweet and the negative verdict ('misleading') notes have lower credibility than the second tweet, then the first tweet is more accurate. Formally, if two tweets have a one-to-one mapping $h : In(t_1) \rightarrow In(t_2)$ with $|verdict(n, t_1)| = |verdict(h(n), t_2)|$, such that $NC(n) >= NC(h(n)) \forall (n, t_1) \in In^+(t_1)$ and $NC(m) <= NC(h(m)) \forall (m, t_1) \in In^-(t_1)$, then $A(t_1) >= A(t_2)$.

The next four axioms define the different factors that affect the note's credibility. The next axiom defines the relation between note and the ratings it receives.

Axiom 3 (Notes with higher credibility get more 'helpful' ratings): Consider two notes n_1 and n_2 with identically trustworthy rating and writing egonetworks and with equal verdicts to equally accurate tweets t_1 and t_2. If for the first note, more 'helpful' ratings have been given than the second note, then the first note has higher credibility. Formally, if two notes n_1 and n_2 are such $TW(u_1) = TW(u_2)$, $A(t_1) = A(t_2)$, $verdict(n_1, t_1) = verdict(n_2, t_2)$, and if there is a one-to-one mapping $h : In(n_1) \rightarrow In(n_2)$ with $TR(u) = TR(h(u))$ and $helpfulness(u, n_1) >= helpfulness(h(u), n_2) \forall (u, n_1) \in In(n_1)$, then $NC(n_1) >= NC(n_2)$.

The next axioms establishes the relationship between note's credibility and user trustworthiness.

Axiom 4 (Notes with higher credibility get 'helpful' ratings from more trustworthy users): Consider two notes n_1 and n_2 with identical and identically trustworthy writing egonetworks and give equal verdicts to equally accurate tweets t_1 and t_2. If the first note has been rated 'helpful' by users who are more trustworthy in rating notes and 'not helpful' by users who are less trustworthy in rating notes, then the first note has a higher credibility. Formally, if two notes n_1 and n_2 are such $TW(u_1) = TW(u_2)$, $A(t_1) = A(t_2)$, $verdict(n_1, t_1) = verdict(n_2, t_2)$, and if there is a one-to-one mapping $h : In(n_1) \rightarrow In(n_2)$ with $helpfulness(u, n_1) = helpfulness(h(u), n_2)$ such that $TR(u) >= TR(h(u)) \forall (u, n_1) \in In^+(n_1)$ and $TR(u) <= TR(h(u)) \forall (u, n_1) \in In^-(n_1)$, then $NC(n_1) >= NC(n_2)$.

The next axiom looks at who wrote the note and their impact on note's credibility.

Axiom 5 (Notes with higher credibility are written by users more trustworthy in writing notes): Consider two notes n_1 and n_2 with identical and identically trustworthy rating egonetwork and with equal verdicts to equally accurate tweets t_1 and t_2. The note written by a user with higher trustworthiness in writing notes has higher credibility. Formally, if two notes n_1 and n_2 are such $TW(u_1) >= TW(u_2)$, $A(t_1) = A(t_2)$, $verdict(n_1, t_1) = verdict(n_2, t_2)$, and if there is a one-to-one mapping $h : In(n_1) \rightarrow In(n_2)$ with $helpfulness(u, n_1) = helpfulness(h(u), n_2)$ and $TR(u) = TR(h(u)) \forall (u, n_1) \in In(n_1)$, then $NC(n_1) >= NC(n_2)$.

The next axiom defines the relation between a tweet's accuracy and the note's credibility.

Axiom 6 (Notes with higher credibility give verdicts closer to tweet accuracy): Consider two notes n_1 and n_2 with identically trustworthy rating and writing egonetworks. The note with the verdict closer to the tweet accuracy has higher credibility. Formally, if two notes n_1 and n_2 are such $TW(u_1) = TW(u_2)$, if there is a one-to-one mapping $h : In(n_1) \rightarrow In(n_2)$ with $helpfulness(u, n_1) = helpfulness(h(u), n_2)$ and $TR(u) = TR(h(u)) \forall (u, n_1) \in In(n_1)$, if $|verdict(n_1, t_1) - A(t_1)| <= |verdict(n_2, t_2) - A(t_2)|$, then $NC(n_1) >= NC(n_2)$.

The next two axioms establish user's trustworthiness. The next axiom defines the relation between users and and the notes written by them.

Axiom 7 (Users more trustworthy in writing notes write notes with high credibility.): For two users who have written an equal number of notes, if one's notes have a higher credibility than the credibility of the notes written by the other user, then the former user has higher trustworthiness in writing notes. Formally, if two users have a one-to-one mapping $h : Out_n(u_1) \rightarrow Out_n(u_2)$ such that $|Out_n(u_1)| = |Out_n(u_2)|$ and $NC(n) >= NC(h(n)) \forall n \in Out_(u_1)$, then $TW(u_1) >= TW(u_2)$.

The next axiom defines the relation between users' rating trustworthiness and the ratings that they give.

Axiom 8 (Users more trustworthy in rating notes give helpfulness ratings scores closer to credibility of notes.): For two users who have rated an equal number of notes, the user whose notes have ratings closer to their credibility scores is more trustworthy in rating notes. Formally, if two users have a one-to-one mapping $h : Out_r(u_1) \rightarrow Out_r(u_2)$ such that $|Out_r(u_1)| = |Out_r(u_2)|$ and $|helpfulness(u_1, n) - NC(n)| <= |helpfulness(u_1, h(n)) - NC(h(n))|$, then $TR(u_1) >= TR(u_2)$.

HawkEye Formulation

Based on the above axioms, we define the formulation for all quality scores such that they satisfy the axioms. Other ways to define the quality scores which satisfy the axioms are feasible as well, which we will explore in future work. The presently proposed formulation is defined below.

• *Trustworthiness of user in rating notes:* Trustworthiness of a user in rating notes is calculated as the average of the closeness between the rating and credibility for all the notes rated by the user. We formulate it as :

$$TR(u) = \frac{\sum_{(u,n) \in Out_r(u)} \left(1 - \frac{|helpfulness(u,n) - NC(n)|}{2}\right)}{|Out_r(u)|}$$

where, $helpfulness(u, n)$ refers to the rating given by user u to note n and $Out_r(u)$ refers to all the ratings given by the user.

• *Trustworthiness of user in writing notes:* Trustworthiness of a user in writing notes is calculated as the average of the credibility for all the notes written by the user and is formulated as :

$$TW(u) = \frac{\sum_{n \in Out_n(u)} NC(n)}{|Out_n(u)|}$$

where, $Out_n(u)$ refers to all the notes written by user u and $Out_t(u)$ refers to all the tweets written by user u

• *Credibility of a note:* Credibility of a note is calculated as a function (aggregation) of three components. The first component is the weighted average of the (helpfulness) ratings received by a note, where the weight is the trustworthiness of the user rating the notes. The second component is the trustworthiness in writing notes of the user who wrote the note. The third component is the closeness of the note's verdict to the accuracy of the tweet the note is written for. We formulate it as :

$$NC(n) = \frac{1}{3}\Big(\lambda_1 \frac{\sum_{(u,n)\in In(n)} TR(u)\cdot helpfulness(u,n)}{|In(n)|} + \lambda_2 TW(u) + \lambda_3\left(1-|A(t)-verdict(n,t)|\right)\Big)$$

where $In(n)$ refers to all the ratings received by note u and $verdict(t)$ refers to the note n's classification for tweet t. We define $\lambda_1, \lambda_2, \lambda_3 \in [0,1]$ as constants which can be tuned as hyperparameters.

• *Accuracy of a tweet:* The accuracy of a tweet is calculated as the weighted average of the verdicts written for the tweet, where the weights are the credibilities of the respective notes.

$$A(t) = \frac{\sum_{(n,t)\in In(t)} NC(n)\cdot verdict(n,t)}{|In(t)|}$$

where $In(t)$ refers to all the notes received by tweet t.

It can be verified that the above formulation satisfies the eight axioms presented in the previous subsection. The formal proof is eliminated due to lack of space.

Addressing the cold start problem

The cold-start problem refers to when elements in a system have had very little interactions with other items in the system, making it very difficult to measure their true quality because of insufficient information. For Birdwatch, we might face the cold start problem to estimate the credibility of notes (when there are very few ratings given to the note), the accuracy of a tweet (in case it has fewer notes) and the trustworthiness of users (who have rated or written very few notes) [31]. For example, it would be wrong to classify an accurate tweet as inaccurate if it only has a couple of notes (by malicious actors) marking it as 'misleading'. Similarly, a user with very few but highly accurate ratings may be a malicious user camouflaging itself or it may be a benign user [32, 33]. Cold start needs to be addressed to distinguish these cases.

We address the cold start issue by adding Laplace smoothing to each of the quality scores. We introduce smoothing parameters $\alpha_1, \beta_1, \gamma_1, \delta_1$ which are pseudo counts whereas $\mu_r, \mu_w, \mu_g, \mu_t$ are the prior beliefs or default scores for trustworthiness of new users in rating, trustworthiness of new users in writing, credibility of new notes and accuracy of new tweets, respectively. The smoothing parameters tune the relative importance of prior — the lower (higher) the value of priors, the more (less) the quality scores depend on that component. The resulting equations are:

$$TR(u) = \frac{\sum_{(u,n)\in Out_r(u)} \left(1 - \frac{|helpfulness(u,n) - NC(n)|}{2}\right) + \alpha_1\mu_r}{|Out_r(u)| + \alpha_1}$$

$$TW(u) = \frac{\sum_{n\in Out_n(u)} NC(n) + \beta_1\mu_w}{|Out_n(u)| + \beta_1}$$

$$NC(n) = \frac{1}{3}\Big(\lambda_1 \frac{\sum_{(u,n)\in In(n)} TR(u)\cdot helpfulness(u,n) + \gamma_1\mu_g}{|In(n)| + \gamma_1} + \lambda_2 TW(u) + \lambda_3\left(1-|A(t)-verdict(n,t)|\right)\Big)$$

$$A(t) = \frac{\sum_{(n,t)\in In(t)} NC(n)\cdot verdict(n,t) + \delta_1\mu_t}{|In(t)| + \delta_1}$$

With the above formulation that incorporates cold start, if a user has written very few notes, its trustworthiness will be closer to the default beliefs, thus preventing the first few notes from having much impact on its trustworthiness. As the user writes more notes, its trustworthiness score moves towards the average of credibility of notes. The same logic applies to the other metrics as well, where we addressed cold start.

These equations satisfy the axioms as well. The Laplace smoothing terms are constants, so have no impact while proving that the formulation satisfies the axioms.

The HawkEye algorithm. Having formulated the intrinsic metrics above, we now describe how the HawkEye algorithm computes the quality scores. We initialize the TR, TW, NC and A scores of all users, notes and tweets to the highest possible value 1. The pseudo counts $\alpha_1, \beta_1, \gamma_1, \delta_1$ are also initialized to 1 and the priors $\mu_r, \mu_w, \mu_g, \mu_t$ are set as the mean scores of TR, TW, NC and A respectively. The algorithm updates each of the values iteratively, so let TR^t, TW^t, NC^t, A^t be vectors representing the trustworthiness in rating of all users, trustworthiness in writing, credibility of all notes and accuracy of all tweets at the end of iteration t. We then iteratively update the scores using the above equations. In each iteration, all the scores in an iteration t are updated using the scores from the previous iteration $t-1$. The iterations are done until convergence, i.e., when all scores change minimally in two consecutive iterations, where ϵ (= 0.001) is the acceptable error bound.

Ranking notes with the HawkEye metric. Here we explain how the HawkEye metrics can be used to rank notes in Birdwatch. Recall that Birdwatch currently uses the helpfulness ratio measure to rank notes (please refer to the Data Section for details). Following that, in HawkEye, we define notes that receive at least 5 ratings and have a minimum credibility threshold τ as the set of '*most credible notes*'. These notes are ranked according to their credibility. We posit that this ranking

system is more robust against adversarial attacks compared to the current system being used by Birdwatch.

Attacker's Goals and Capabilities

The **primary goal of the attackers** is to manipulate the set of top-ranked notes for a target tweet. Specifically, the attacker aims to promote a random note (which is not already top-ranked) to be among the top-ranked notes according to the ranking metric. This attack can involve two scenarios. If the size of the set of top-ranked notes is less than k, a the attacker would aim to *insert* the randomly chosen target note in the top-ranked notes set. If the size of the set of top-ranked notes is atleast k, the attacker would *replace* one of the existing top-ranked notes with the target note. We describe the two attacks as insertion and replacement attacks, respectively. To study the most challenging attack setting, we set k = 1, i.e., the attacker's goal is to make the target note as the topmost ranked note of the tweet. The task for the attacker naturally becomes easier when the value of k is increased.

The attacker has the following **capabilities**:

• *Account creation:*The attacker can create (multiple) accounts on the Birdwatch platform.

• *Rating notes:* The attacker can give ratings (helpful or not helpful) to existing notes. One account can only rate a note once. The attacker can not write new notes.

• *White box attack:* The attacker has access to the Birdwatch data, which is already publicly available. We also assume that the attacker knows the thresholds being used by the reputation system, which is feasible as Birdwatch's entire code is publicly available.

We quantify a **reputation system's robustness** as follows: *How many accounts does an attacker need to promote a non top-ranked note to become the topmost ranked note?* Higher numbers mean that the reputation system is more robust.

The top ranked note for the HawkEye metric is the note with the highest credibility score and at least 5 ratings. On the other hand, the top ranked note for the existing Birdwatch system would be the note with at least 5 ratings and has the highest helpfulness ratio among the notes marked as 'currently rated helpful' (CRH).

Experimental Evaluation

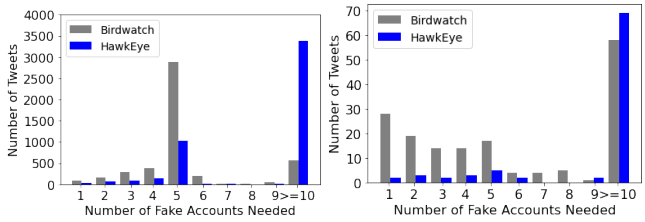
In this section, we conduct the following experiments:

• **Robustness against attack.** We evaluate the robustness of Birdwatch and HawkEye against the adversarial attack. Recall that the goal of the attacker is to make a random note the top ranked note.

• **Sensitivity to hyperparameter.** We evaluate how sensitive HawkEye is to the minimum credibility threshold τ parameter.

• **Misinformation tweet detection.** We evaluate how accurate Birdwatch and HawkEye are in classifying tweets as accurate or misleading.

Experiment details. The weighing constants λ_1, λ_2 and λ_3 are set to 0.1. In HawkEye, we set $\tau = 0.02$ and $\epsilon = 0.001$.



(a) Insertion Attack Distribution (b) Replacement Attack Distribution

Fig. 2: Adversarial attack comparison for Birdwatch and HawkEye ($\tau = 0.02$) systems.

Experiment 1: Robustness against attack.

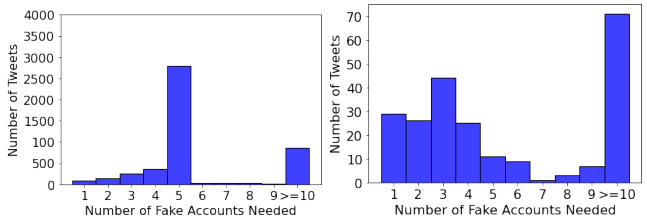
This experiment answers **research questions 1 and 2**. We compare the HawkEye metric's robustness against the adversary to the Birdwatch metric's robustness via simulations.

Given a reputation system R, for every tweet, we randomly select a note n which is not already a top-ranked note according to R. Then we iteratively add fake accounts, which give fake 'helpful' ratings to note n and 'not helpful' ratings to notes currently ranked at the top by R. We find the minimum number of fake accounts needed for the note n to become the top ranked note and meet all the criteria by R. This process is repeated for all the tweets. We only allow a maximum of 10 fake accounts to be used by the attacker (for computational reasons).

The above iterative process generates a distribution of the number of fake accounts needed to boost a random note to top-ranked in R's reputation system. We generate this distribution for both the HawkEye metric and the Birdwatch metric. The resulting plot is shown in Figure 2. We split the plot into two cases. First is the *insertion attack* (Figure 2(a)), when prior to the attack there was no note that satisfied the system R's criteria, in which case the target note is promoted to be the highest ranked note. The second case is *replacement attack* (Figure 2(b)), when prior to the attack there is already a note that occupies the top rank position (and satisfies R's criteria). Here the target note replaces the current top ranked note as the new top ranked note after the attack.

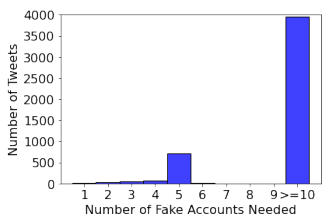
First, we notice that there are more insertion attack cases than replacement attack cases. Next, we see that for the Birdwatch metric, the peak in the insertion attack plot is at 5 fake accounts, which, you may recall, is the minimum number of ratings that are required for a note to quality as a CRH note. Third, we find that number of fake accounts needed in HawkEye is significantly higher than that in Birdwatch. The curves for HawkEye have shifted more towards the right compared to Birdwatch, which indicates higher robustness of the HawkEye system.

To statistically compare the distributions, we run a paired-t test to assess if the samples for the Birdwatch metric and HawkEye the metric have identical expected values. We find that in both the insertion and the replacement attack case, the distributions are statistically different with p-values < 0.05. The expected value of HawkEye system is higher than the Birdwatch system. This proves that the proposed HawkEye

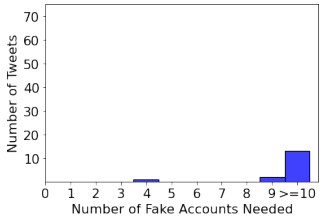


(a) Insertion Attack, $\tau = 0.01$

(b) Replacement Attack, $\tau = 0.01$



(c) Insertion Attack, $\tau = 0.03$



(d) Replacement Attack, $\tau = 0.03$

Fig. 3: Adversarial attack plots for HawkEye system with different parameters.

system has a higher robustness than the Birdwatch system.

Credibility Threshold τ	Insertion p-value	Replacement p-value
0.01	1.08e-17	0.97
0.02	0.0	7.27e-07
0.03	0.0	0.01

TABLE I: Paired t-test p-values comparison.

Experiment 2: Sensitivity to τ.

Next, we evaluate the sensitivity of the τ parameter in HawkEye, i.e., the minimum credibility threshold. Naturally, as τ increases, so does the difficulty of breaking the HawkEye system. In addition to τ = 0.02, here we test for two more τ values of 0.01 and 0.03. The plots for both the insertion and replacement attacks are shown in Figure 3. As expected, the curve shifts towards the right for the higher τ value. Similar to the previous experiment, we also compare the curves with the Birdwatch system via paired t-test and report the p-values in Table I. We find that both the insertion attack curves have a statistically significant p-value (< 0.05). On the other hand, only the replacement attack plot of τ = 0.03 is significantly higher than Birdwatch.

Experiment 3: Misinformation tweet detection performance.

This experiment answers **research question 3**. Here we compare the performance of Birdwatch's and HawkEye's reputation system in classifying tweets as accurate or misleading.

As the tweets do not have ground-truth labels, we seek the help of three non-author annotators to label the tweets as misinformation or accurate. We select 500 tweets from the entire set of tweets to be labeled. To help them make a more informed decision, we also supply several attributes around each tweet including the (1) tweet text and URL, (2) the user handle, display name, bio, user follower count and verification status, and (3) number of likes and retweets. The annotators mark each tweet as either accurate or misleading. A Fleiss Kappa score of 0.51 amongst the 3 annotators shows a moderate inter-rater agreement. The ground-truth label for each tweet is assigned as the majority of the three individual annotations. 124 tweets were assigned the misinformation label.

Metric	Birdwatch	HawkEye (unsupervised)	HawkEye (supervised)
Precision	0.63	0.85	0.79
Recall	0.25	0.74	0.78
F1-score	0.11	0.76	0.78

TABLE II: Misinformation tweet detection performance comparison between Birdwatch and HawkEye.

The predicted labels for the tweets for both the reputation systems, Birdwatch and HawkEye, are determined as follows. For the Birdwatch system, we count the number of helpful notes as the notes that have a helpfulness ratio of at least 0.84 (this threshold is used by Birdwatch). For each tweet, if the number of helpful notes that labeled the tweet as misleading are more than or equal to the number of helpful notes that labeled the tweet as not-misleading, we say Birdwatch classifies the tweet as misleading. We use this technique as Birdwatch does not define a standard way for tweet labeling.

Analogous to the above method, we employ a similar unsupervised method, to detect misinformation tweets for the HawkEye system. For each tweet, from the notes written for the tweet, we select notes having a credibility of at least 0.02. You may recall these credible notes by HawkEye. Among these notes, if the number of notes that labeled the tweet as misleading are more than or equal to the number of notes that labeled the tweet as not misleading, HawkEye classifies the tweet as misleading.

Furthermore, for the HawkEye system, we also employ a supervised learning technique that trains a model to detect misinformation tweets. We represent each tweet t as a feature vector of its accuracy scores across multiple runs of HawkEye for different combinations of values of the weighing constants $\lambda_1, \lambda_2, \lambda_3$ and of the smoothing parameters $\alpha_1, \beta_1, \gamma_1, \delta_1$ i.e. $A(t|\alpha_1, \beta_1, \gamma_1, \delta_1, \lambda_1, \lambda_2, \lambda_3) \forall (\alpha_1, \beta_1, \gamma_1, \delta_1, \lambda_1, \lambda_2, \lambda_3) \in \mathcal{C}$ where $\mathcal{C}$ is the set of all parameter combinations. This is done to overcome biases that may creep in due to a single parameter setting. We choose the smoothing parameters values from $0, 1, 2$ and the weighing constants λ_i from $0, 0.5, 1$ and end up with $3^7 = 2187$ combinations. For each tweet t in the 500 labeled tweet, we get the $A(t)$ values for all the parameter combinations, which forms a 2187-dimension vector for every tweet. Using these vectors, we train a random forest classifier and conduct 10-fold cross validation on the 500 tweets. Average numbers across the ten folds are calculated.

All the results are shown in Table II. We evaluate the class-weighted average precision, recall and F1-scores.

The results in the table show that the HawkEye (unsupervised) algorithm performs better than the Birdwatch technique in identifying inaccurate tweets. This holds true across all three performance metrics. Between HawkEye unsupervised and HawkEye supervised, we see that the supervised version performs slightly better than the unsupervised one in terms of recall and F-1 score.

Discussion and Conclusion

In this work, we showed that HawkEye is more robust then Birdwatch's current system against adversaries who try to manipulate note rankings. The work has some limitations. First, the proposed system does not use the text information present in the notes. This is because the current note texts are mostly sparse. Textual feature can be included in the future when the text content is more useful. Second, we only study one type of adversarial attack. Robustness to other attacks remains to be an open challenge.

References

[1] K. Shu, A. Sliva, S. Wang, J. Tang, and H. Liu, "Fake news detection on social media: A data mining perspective," *ACM SIGKDD explorations newsletter*, 2017.

[2] S. Vosoughi, D. Roy, and S. Aral, "The spread of true and false news online," *Science*, 2018.

[3] N. Grinberg, K. Joseph, L. Friedland, B. Swire-Thompson, and D. Lazer, "Fake news on twitter during the 2016 us presidential election," *Science*, 2019.

[4] S. Kumar, R. West, and J. Leskovec, "Disinformation on the web: Impact, characteristics, and detection of wikipedia hoaxes," in *WWW*, 2016.

[5] S. Jiang, M. J. Metzger, A. J. Flanagin, and C. Wilson, "Modeling and measuring expressed (dis)belief in (mis)information," in *ICWSM*, 2020.

[6] E. S. Radwan and A. Radwan, "The spread of the pandemic of social media panic during the covid-19 outbreak," 2020.

[7] M. S. Islam, T. Sarkar, S. H. Khan, A.-H. M. Kamal, S. M. Hasan, A. Kabir *et al.*, "Covid-19–related infodemic and its impact on public health: A global social media analysis," *AJTMH*, vol. 103, 2020.

[8] E. Kaufman, "The social media, the mass media, violence and democracy. the arena for institutional weakening and citizens lack of confidence," 2018.

[9] S. Kumar and N. Shah, "False information on web and social media: A survey," *arXiv:1804.08559*, 2018.

[10] K. Sharma, F. Qian, H. Jiang, N. Ruchansky, M. Zhang, and Y. Liu, "Combating fake news: A survey on identification and mitigation techniques," *ACM TIST*, 2019.

[11] F. Alam, S. Shaar, F. Dalvi, H. Sajjad *et al.*, "Fighting the covid-19 infodemic: modeling the perspective of journalists, fact-checkers, social media platforms, policy makers, and the society," *arXiv:2005.00033*, 2020.

[12] J. M. Burkhardt, *Combating fake news in the digital age*. American Library Association, 2017, vol. 53, no. 8.

[13] N. Micallef, B. He, S. Kumar, M. Ahamad, and N. Memon, "The role of the crowd in countering misinformation: A case study of the covid-19 infodemic," *IEEE BigData*, 2020.

[14] A. L. Wintersieck, "Debating the truth: The impact of fact-checking during electoral debates," *American politics research*, 2017.

[15] R. J. Sethi, "Crowdsourcing the verification of fake news and alternative facts," in *ACM Hypertext*, 2017.

[16] J. Kim, B. Tabibian, A. Oh, B. Schölkopf, and M. Gomez-Rodriguez, "Leveraging the crowd to detect and reduce the spread of fake news and misinformation," in *ACM WSDM*, 2018.

[17] M. R. Pinto, Y. O. de Lima, C. E. Barbosa, and J. M. de Souza, "Towards fact-checking through crowdsourcing," in *IEEE CSCWD*, 2019.

[18] Y. Wang, W. Yang, F. Ma, J. Xu, B. Zhong, Q. Deng, and J. Gao, "Weak supervision for fake news detection via reinforcement learning," in *AAAI*, 2020.

[19] R. Denaux, F. Merenda, and J. Manuel, "Towards crowdsourcing tasks for accurate misinformation detection."

[20] F. Tchakounté, A. Faissal, M. Atemkeng, and A. Ntyam, "A reliable weighting scheme for the aggregation of crowd intelligence to detect fake news," *Information*, vol. 11, no. 6, p. 319, 2020.

[21] S. Shabani and M. Sokhn, "Hybrid machine-crowd approach for fake news detection," in *IEEE CIC*, 2018.

[22] S. Tschiatschek, A. Singla, M. Gomez Rodriguez, A. Merchant, and A. Krause, "Fake news detection in social networks via crowd signals," in *WWW*, 2018.

[23] S. Kumar, B. Hooi, D. Makhija, M. Kumar, C. Faloutsos, and V. Subrahmanian, "Rev2: Fraudulent user prediction in rating platforms," in *ACM WSDM*, 2018.

[24] S. Rayana and L. Akoglu, "Collective opinion spam detection: Bridging review networks and metadata," in *ACM SIGKDD*, 2015.

[25] G. Wang, S. Xie, B. Liu, and S. Y. Philip, "Review graph based online store review spammer detection," in *IEEE ICDM*, 2011.

[26] M. Jiang, P. Cui, A. Beutel, C. Faloutsos, and S. Yang, "Catchsync: catching synchronized behavior in large directed graphs," in *ACM SIGKDD*, 2014.

[27] S. Pandit, D. H. Chau, S. Wang, and C. Faloutsos, "Netprobe: a fast and scalable system for fraud detection in online auction networks," in *WWW*, 2007.

[28] A. Chetan, B. Joshi, H. S. Dutta, and T. Chakraborty, "Corerank: Ranking to detect users involved in blackmarket-based collusive retweeting activities," in *ACM WSDM*, 2019.

[29] U. Gadiraju, R. Kawase, S. Dietze, and G. Demartini, "Understanding malicious behavior in crowdsourcing platforms: The case of online surveys," in *CHI*, 2015.

[30] F. Daniel, P. Kucherbaev, C. Cappiello, B. Benatallah, and M. Allahbakhsh, "Quality control in crowdsourcing: A survey of quality attributes, assessment techniques, and assurance actions," *ACM CSUR*, 2018.

[31] B. Hooi, H. A. Song, A. Beutel, N. Shah, K. Shin, and C. Faloutsos, "Fraudar: Bounding graph fraud in the face of camouflage," in *ACM SIGKDD*, 2016.

[32] E.-P. Lim, V.-A. Nguyen, N. Jindal, B. Liu, and H. W. Lauw, "Detecting product review spammers using rating behaviors," *ACM CIKM*, 2010.

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