

Semi-Supervised Correction of Biased Comment Ratings

Abhinav Mishra
Yahoo! Labs Bangalore
abhinavm@yahoo-inc.com

Rajeev Rastogi
Yahoo! Labs Bangalore
rrastogi@yahoo-inc.com

ABSTRACT

In many instances, offensive comments on the internet attract a disproportionate number of positive ratings from highly biased users. This results in an undesirable scenario where these offensive comments are the top rated ones. In this paper, we develop *semi-supervised learning* techniques to correct the bias in user ratings of comments. Our scheme uses a small number of comment labels in conjunction with user rating information to iteratively compute *user bias* and *unbiased ratings* for unlabeled comments. We show that the running time of each iteration is linear in the number of ratings, and the system converges to a unique fixed point. To select the comments to label, we devise an *active learning* algorithm based on *empirical risk minimization*. Our active learning method incrementally updates the risk for neighboring comments each time a comment is labeled, and thus can easily scale to large comment datasets. On real-life comments from Yahoo! News, our semi-supervised and active learning algorithms achieve higher accuracy than simple baselines, with few labeled examples.

Categories and Subject Descriptors

H.2.8 [Information Systems]: Database Applications - Data mining

General Terms

Algorithms, Experimentation

Keywords

Semi-supervised learning, active learning, iterative technique, bias

1. INTRODUCTION

With the proliferation of web applications such as blogs, wikis, social networking sites, etc., users can not only consume news content but also share their opinions. On many web sites such as Yahoo! News, a user can post comments, reply to other comments, and even provide ratings to other comments. As [19] notes, user experience on the internet is becoming a shared social experience through discussions, tweets, comments, etc. In fact, [19] reports that around 25% of internet users have commented on a news article.

Copyright is held by the International World Wide Web Conference Committee (IW3C2). Distribution of these papers is limited to classroom use, and personal use by others.

WWW 2012, April 16–20, 2012, Lyon, France.
ACM 978-1-4503-1229-5/12/04.























the only good *** is a dead *** i like dead ***	15			6
if I burn my ***and nobody sees it... did it really burn?	51			23
*** priest are homosexual***	84			62
Its high time we threw all *** out of America	46			12
*** is a savage, backward, intolerant political ***	75			5
reinforces my opinion ... good *** is a dead one.	14			1
It's *** was a pedophile and mass murderers	58			7
Clinton's rating went up and it was confirmed sex...	254			52
Scott, Then you are either a traitor or an idiot	27			8
Khloe is a beast!! She must've had a different daddy.	25			3
photo made her look like someone stuck a corncob up her ***	168			7

Figure 1: Sample abusive comments that receive many thumbs-up ratings.

Depending on the news event, an article can receive either a few or thousands of comments. For a user, it is simply not possible to read and rate all the comments. To address this, web sites such as Yahoo! News provide a number of options to order the comments for an article. These include sorting by time, most replied, top rated, popular now, etc. Of these measures, top rated and popular now do require some algorithmic computation on the ratings provided by users. For instance, a user gives a comment a thumbs-up rating to indicate a positive vote and a thumbs-down to signal a negative vote. The top rated comments are then the ones with the highest fraction of thumbs-up ratings. Thus, crowdsourced ratings are used to measure the quality of a comment, and the whole system is moderated using crowdsourcing.

A comment from a user represents his/her view on a topic. On sensitive topics, however, comments do appear to be highly biased. One study [7] found that categories like war, crime, law, politics, society, and many others have a higher fraction of comments that get filtered. On many topics such as war and immigration, we often see hate speech directed at the people of another community. In Figure 1, we show examples of this where the content is obviously abusive. To a neutral (unbiased) person, such content would appear very offensive. Clearly, such comments should either be hidden or deleted.

Web sites do have a policy against posting abusive content, hate speech, spam, etc. Currently, most of the automated systems rely on some form of human activity to iden-

tify offensive content. Such human activity includes flagging as spam, giving thumbs-down to comments, etc. If the number of spam flags or thumbs-down ratings is high then these comments are removed.

In [14], it was shown that crowd-based moderation is indeed effective (on Slashdot) in a controlled environment. However, due to human bias, offensive comments may not always attract many thumbs-down ratings. Even editors and moderators acknowledge that there are some gray areas, and people’s perceptions of what constitutes distasteful material can vary [8]. To make matters worse, such comments often attract a large number of thumbs-ups (or thumbs-downs) from other biased users who easily outnumber the unbiased ones. For example, in Figure 1, a majority of users have given the comments a thumbs-up rating. Since such comments receive many thumbs-ups, they appear as the top rated comments, which is clearly undesirable. Such comments should be hidden, or at the very least, should not appear at the top.

The main problem here lies with the metric that is used to compute top rated comments. Simple statistical measures such as fraction of thumbs-up are not sufficiently equipped to identify the bad comments or reduce their ratings. They do not factor bias while calculating the rating for a comment. Ideally, the opinion of a highly biased user should be given less weight, i.e., user reputation should be considered as well in the calculation. Similar intuition underlies modern ranking algorithms like HITS [12] and PageRank [4], where the outlinks from a highly ranked page are considered more importance.

In this paper, we propose unsupervised, semi-supervised, and active learning algorithms for correcting the bias in comment ratings provided by users; thus, our algorithms compute the unbiased ratings of comments. In our work, we only rely on the structural aspects, i.e., the user-comment graph containing who-rated-what. We do not make use of the content of comments.

Our main contributions are as follows:

- 1) We give mutually-recursive definitions of user bias and unbiased comment rating in terms of one another leveraging the user-comment graph (Section 3). We use fixed-point iteration to solve the mutually-recursive equations of bias and rating in time that is linear in the number of ratings. We prove that our iterations converge to a unique fixed-point. We also present an analytical solution to our problem and show a connection with randomized HITS [27].
- 2) In many instances, ratings from biased users easily outnumber those from others. To determine the unbiased ratings in these cases, we use semi-supervision (Section 4). Specifically, we obtain labels from human editors that reflect the ground truth for a small number of comments. We use these comment labels in conjunction with the rating information provided by users to iteratively compute the unbiased ratings for unlabeled comments.
- 3) Since labeling comments is expensive, we propose a scalable active learning algorithm to select the comments to label (Section 5). In each step, our active learning algorithm greedily selects the comment with the lowest expected risk as the next comment to label, and incrementally updates the risk for neighboring comments. In the computation of

expected error, we use the true rating computed for a comment (by our semi-supervised method) to approximate the probability of a thumbs up. We devise efficient algorithms for incrementally updating the ratings and expected errors of comments in the neighborhood of a newly labeled comment.

- 4) We conduct an extensive experimental study with real-life comments from Yahoo! News (Section 7). Even without supervision, our algorithm significantly outperforms simple baselines. Furthermore, the accuracy of our predicted ratings increases as more comments are labeled. Finally, our active learning scheme achieves high accuracy with 5 times fewer labeled examples compared to random labeling schemes.

2. SYSTEM MODEL

We consider a model where users post comments on a news article expressing their views. Other users have a choice to show their like/dislike through a thumbs-up/thumbs-down rating. A user is not allowed to give more than one rating to a comment. We model this system using a bipartite graph G , where users are on one side and comments are on the other side. A directed edge from a user to a comment represents that the user has given a thumbs-up or thumbs-down. Let w_{ij} denote the rating given by user i to comment j . Here, $w_{ij} \in \{-1, +1\}$, where $w_{ij} = +1$ corresponds to a thumbs-up rating and $w_{ij} = -1$ means a thumbs-down rating. Note that a user will have only outgoing edges and likewise, a comment will have only inlinks.

A user may be biased on certain topics, which makes his/her ratings biased as well. However, the user may behave normally on other topics. This phenomenon of *topicality* is also pointed out by [8], especially on many sensitive topics such as immigration. We say that a user is biased with respect to a topic if his/her opinion deviates significantly from the correct opinion on the comments belonging to that topic. Therefore, we have different bias scores for a user for different topics.

The standard method used to compute the rating of a comment j is to take the mean of all ratings ($\text{avg}_i\{w_{ij}\}$) the comment receives (at times, over a certain period). However, such a method does not factor in bias. Our objective in this paper is to develop techniques which also factor in bias while computing the rating of a comment. We refer to such a rating as the *unbiased rating*. We compute the bias for users and unbiased ratings for comments separately for each topic.

3. UNSUPERVISED TECHNIQUE

In this section, we present an unsupervised algorithm to compute the bias and unbiased rating of a comment. We show a mutually recursive relationship between bias and unbiased rating, i.e., we write bias as a function of unbiased rating and unbiased rating as a function of bias. We use fixed-point iteration to solve this mutually recursive equation. Later, we propose a semi-supervised framework to further improve this approach by allowing some comments to be explicitly labeled.

3.1 Bias definition

As mentioned earlier, a user is considered to be biased if his/her ratings deviate considerably from the correct ratings.

Thus, intuitively, if $r(j)$ is the unbiased rating of comment j , then the function $|w_{ij} - r(j)|$ is a good choice for capturing the bias of user i on comment j . If the gap between the user rating and unbiased rating is more, then the function $|w_{ij} - r(j)|$ will attain a high value reflecting a high level of bias. On the other hand, if the gap is small, then the bias will be low. Recall that a user gives either a thumbs-up or thumb-down rating. Therefore, $w_{ij} \in \{-1, 1\}$. In contrast, we allow the unbiased rating $r(j)$ to take any value between -1 and 1 ; thus, $r(j) \in [-1, 1]$.

The function $|w_{ij} - r(j)|$ can be equivalently written as $(1 - w_{ij} \cdot r(j))$. We use this latter equivalent formulation throughout the paper. The bias of a user i on comment j after normalization is given as:

$$bias_j(i) = \frac{1 - w_{ij} \cdot r(j)}{2} \quad (1)$$

Equation (1) computes bias only for one comment. In order to compute the bias of a user over all comments he/she has rated on a given topic, we simply take the mean. Therefore, bias can be viewed as the expected error a user is likely to make on a comment. Let $d^o(i)$ denote the number of ratings given by a user i on a topic. Then, bias is defined as:

$$bias(i) = \frac{1}{2 \cdot d^o(i)} \sum_{j:i \rightarrow j} (1 - w_{ij} \cdot r(j)) \quad (2)$$

Observe that if a user gives many irrational ratings ($w_{ij} \neq r(j)$), then he/she will have a high bias. Also, $bias \in [0, 1]$, where 0 signifies that a user is unbiased and 1 indicates that his/her opinion is highly biased. Our definition of bias has a remarkable similarity with the hubs vector of randomized HITS [27] (see Section 6.2).

In our framework, bias is the expected error a user is likely to make while rating a comment. Related approaches [13, 17] that address bias in user ratings, and their follow-up work, define bias as the propensity to give high or low ratings. Roughly, the bias of a user is viewed as the average inflation or deflation in ratings by the user. However, such a definition is not applicable in our comment rating environment since a user can artificially keep his/her bias close to zero, while in reality, it is very high. For example, consider a user who always gives ratings opposite to the unbiased ratings. Using our method, such a user will have a high bias score, whereas with the schemes of [13, 17], it is possible for a user to keep his/her bias close to zero.

There is another noteworthy difference which makes the above approaches unsuitable for computing the bias. Consider a user who likes a particular movie. Whenever a good comment about the movie appears, he/she is likely to give a thumbs-up rating, and if a negative comment appears, he/she is expected to give a negative rating. This intuition conforms with our definition of bias. However, with the other definitions, the user is expected to give either high ratings to both positive and negative comments about the movie, or low ratings to both.

3.2 Unbiased rating definition

We compute the unbiased rating of a comment by factoring user bias. If a comment receives a rating from a highly biased user, then we give a lower weight to that user. Intuitively, we ignore a user's opinion if it is frequently wrong. If a user i is biased, then the quantity $(1 - bias(i))$ is the confidence with which he/she gives the correct rating. For

instance, if $(1 - bias(i)) = 0$, then the user is very likely to give an incorrect rating. We modify the rating by factoring bias as:

$$w'_{ij} = w_{ij}(1 - bias(i)) \quad (3)$$

The above formulation ensures that if a user is unbiased, then his/her rating remains unchanged. However, if he/she is biased, say $bias(i) = 0.9$, then we reduce the rating to 0.1 fraction. To compute the unbiased rating r , we simply take the mean of all the modified ratings as follows:

$$r(j) = \frac{1}{d^i(j)} \sum_{i:i \rightarrow j} w'_{ij} \quad (4)$$

Here, $d^i(j)$ denotes the number of incoming links (ratings) comment j receives. Note that $r(j) \in [-1, 1]$. Like bias, unbiased rating is similar to the authority vector of randomized HITS [27] (see Section 6.2).

Observe that in Equation (4), we divide by the number of inlinks. Instead of this simple mean, a weighted mean could be seen as a better option, i.e., dividing by $\sum_i (1 - bias(i))$. However, this has a major drawback. For example, consider two heavily biased users with $bias = 0.99$. Let the users give ratings with a score 1 to the same comment. Then, the rating with weighted mean is still 1. However, with simple mean, we get a rating of 0.01 (close to neutral). Since the two users are heavily biased, their opinion cannot be trusted, and so giving something close to a neutral rating is the best option – the weighted mean does not provide this.

3.3 Bias and unbiased rating computation

We use fixed-point iteration to solve the mutually-recursive equations of bias and rating. In Section 4.2, we show that this system converges to a unique fixed point, i.e., converges to a unique solution irrespective of initial conditions. Let $bias^t(i)$ and $r^t(j)$ denote the bias and rating, respectively, in iteration t . Then, $r^{t+1}(j)$ is computed as follows:

$$r^{t+1}(j) = \frac{1}{d^i(j)} \sum_{i:i \rightarrow j} w_{ij}(1 - bias^t(i)) \quad (5)$$

Similarly, we compute $bias^{t+1}(i)$ as:

$$bias(i)^{t+1} = \frac{1}{2 \cdot d^o(i)} \sum_{j:i \rightarrow j} (1 - w_{ij} \cdot r^{t+1}(j)) \quad (6)$$

3.4 Time complexity

Computing the bias of users, given the unbiased ratings of all comments, requires $O(m)$ time, where m is the number of ratings. Similarly, $O(m)$ time is required to compute the unbiased ratings. Thus, if k is the number of iterations needed to reach fixed point, then the complexity of the algorithm is $O(km)$. Usually the number of ratings provided by users is considerably less than all the possible ratings. This makes the fixed point computation very efficient. Later on, in Section 4.2, we also give a bound on the number of iterations required to guarantee a certain level of accuracy.

4. SEMI-SUPERVISED TECHNIQUE

As discussed earlier, it is possible that a majority of the users who give ratings are biased on a topic. Then, the above formulation may not work, and worse, it is likely to make other opinions look biased. For example, let +1 be

the neutral (correct) view for a comment, and -1 the biased view. If a majority gives the comment -1 , then users who gave $+1$ will be deemed as biased, even though they give correct ratings. In order to address this anomaly, we assume that we know the labels of a few comments apriori. By doing so, we can ensure that unbiased users are not punished even though they are in small numbers. Now, there will be many comments and users who are not a part of the labeled set. By including the available unlabeled set as well for training, we let our semi-supervised algorithm learn ratings from both labeled and unlabeled data.

We use active learning (presented in Section 5) to identify the best comments to label. Unlike traditional graph-based semi-supervised learning [3, 28, 29], we do not make any smoothness assumptions, i.e., neighboring nodes share similar labels, as it is not applicable on a user-comment graph.

4.1 Algorithm

Let L and UL be the two sets of labeled and unlabeled comments, respectively. In general we assume that $|L| \ll |UL|$. We now define notation that will be used later. Let $L(j)$ represent the label of comment j , if $j \in L$. Furthermore, let $d_L^o(i)$ denote the number of labeled ratings available for user i and likewise, $d_{UL}^o(i)$ denote the number of unlabeled ratings. Therefore, $d_L^o(i) + d_{UL}^o(i) = d^o(i)$. We now modify the expressions for unbiased rating and bias. The expression for r is:

$$r(j) = \begin{cases} L(j) & \text{if } j \in L \\ \frac{1}{d^i(j)} \sum_{i:i \rightarrow j} w'_{ij} & \text{otherwise} \end{cases} \quad (7)$$

Bias can also be divided into two types: one from the labeled set and the other from the unlabeled set. The new bias formulation is as follows:

$$\begin{aligned} bias(i) = & \frac{1}{2(\alpha \cdot d_L^o(i) + d_{UL}^o(i))} \left(\alpha \sum_{j:i \rightarrow j, j \in L} (1 - w_{ij} \cdot L(j)) \right. \\ & \left. + \sum_{j:i \rightarrow j, j \in UL} (1 - w_{ij} \cdot r(j)) \right) \end{aligned} \quad (8)$$

Here, the first sum term computes the deviation for user i using labeled comments which he/she has rated. Whereas the second term uses the ratings obtained from the unlabeled set. Since $|L| \ll |UL|$, we give a higher disagreement cost to the labeled data, i.e., we give a higher weight to bias introduced due to the labeled data. The parameter $\alpha > 1$ helps to maintain the relative weight between labeled and unlabeled comments. If the number of labeled comments is really small, then we may choose to have a higher value for α . It is also helpful to choose a large α when a significant fraction of users are biased on a certain topic.

4.2 Algorithm properties

In this section, we show that the difference between two iterations is bounded. Later, this is used to show convergence. We then prove the uniqueness of the system.

4.2.1 Error bound

THEOREM 1. *The difference in bias of a node between two consecutive iterations t and $t + 1$ is bounded by an inverse*

exponential function of t :

$$|bias^{t+1}(i) - bias^t(i)| \leq \left(\frac{1}{2}\right)^t. \quad (9)$$

PROOF. See Appendix B. \square

We use this bound to show convergence. This bound also helps to determine the number of iterations required apriori.

4.2.2 Convergence

For some $\epsilon > 0$, let x be the smallest integer such that $\sum_{k=x}^{\infty} \frac{1}{2^k} < \epsilon$. Here, due to Theorem 1, ϵ represents the maximum difference in the bias values between the iteration x and any iteration after that. Therefore, for any iterations $p, q > x$, we have :

$$|bias^p(i) - bias^q(i)| < \sum_{k=x}^{\infty} \frac{1}{2^k} < \epsilon$$

The above sequence is a Cauchy sequence and thus converges. In practice, we iterate until the difference (L_1 norm) between two consecutive iterations across all nodes becomes small, say δ . It can easily be shown that in order to get within the δ range, we are required to do at most $\log_2 \frac{\eta}{\delta}$ iterations. We omit the proof due to space constraints. In practice, we observed that the algorithm converges in fewer iterations (see Section 7.2.4).

4.2.3 Uniqueness

In this section, we provide a proof (by contradiction) for the uniqueness of the system. Let there be at least two values of bias which satisfy the bias equation. Let $bias^1(i)$ and $bias^2(i)$ be two converged values of node i . Further, let $\delta(i) = bias^1(i) - bias^2(i)$ and $M = \max_p |\delta(p)|$. Also, let i be the node for which we get M . Now, if we can show that M is zero, then uniqueness is proved.

THEOREM 2. *There exists a unique solution to the bias equation.*

PROOF.

$$\begin{aligned} M &= \left| \frac{1}{2(\alpha \cdot d_L^o(i) + d_{UL}^o(i))} \left(\sum_{j:i \rightarrow j, j \in UL} w_{ij}(r^1(j) - r^2(j)) \right) \right| \\ &\leq \left| \frac{1}{2(\alpha \cdot d_L^o(i) + d_{UL}^o(i))} \left(\sum_{j:i \rightarrow j, j \in UL} \left(\frac{1}{d^i(j)} \sum_{k:k \rightarrow j} |bias^2(k) - bias^1(k)| \right) \right) \right| \\ &\leq \left| \frac{1}{2(\alpha \cdot d_L^o(i) + d_{UL}^o(i))} \left(\sum_{j:i \rightarrow j, j \in UL} \left(\frac{1}{d^i(j)} \sum_{k:k \rightarrow j} M \right) \right) \right| \\ &= \frac{M \cdot d_{UL}^o(i)}{2(\alpha \cdot d_L^o(i) + d_{UL}^o(i))} \leq \frac{M}{2} \end{aligned}$$

By definition, M is a positive quantity. So the above inequality only holds when $M = 0$. \square

5. ACTIVE LEARNING

In the semi-supervised approach, we supply labels to some of the comments. We use an active learning based strategy to pick the comments for labeling. Active learning has

been shown to give significant improvements with few labeled points when chosen interactively. See [21] for a detailed survey and references therein.

5.1 Framework

We develop a technique based on empirical risk minimization [20, 29]. In each successive step, we pick a comment to label such that it minimizes the expected risk on the whole data. We assume the true label $L(j)$ for a comment j to be either $+1$ or -1 . For a rating, $r(j)$, computed using the existing labeled and unlabeled data, we define a loss function as:

$$\text{loss}(j) = \begin{cases} (L(j) - r(j))/2 & \text{if } L(j) = 1 \\ (r(j) - L(j))/2 & \text{if } L(j) = -1 \end{cases} \quad (10)$$

This is not the typical 0/1 loss function. Instead we capture the normalized absolute distance between the computed rating $r(j)$, and its true label $L(j)$. Notice that our loss function has exactly the same formulation as bias (see Section 3.1). Now, the expected risk across all the comments $1, \dots, n$ is given by:

$$\text{Risk} = \sum_{i=1}^n \sum_{l=-1, +1} p(L(j) = l) \cdot \text{loss}(j) \quad (11)$$

Since the true label $L(j)$ is unknown, we use $r(j)$ to estimate the probability that $L(j)$ has label l . We ensure that this probability lies in $[0, 1]$ by shifting and scaling $r(j)$.

$$p(L(j) = +1) \approx \frac{(1 + r(j))}{2} \quad (12)$$

Therefore, the estimated expected risk, $\hat{\text{Risk}}$ can be computed as follows:

$$\begin{aligned} \hat{\text{Risk}} &= \sum_{j=1}^n \left(\frac{(1 + r(j))}{2} \cdot \frac{(1 - r(j))}{2} + \frac{(1 - r(j))}{2} \cdot \frac{(1 + r(j))}{2} \right) \\ &= \sum_{j=1}^n \frac{1 - r^2(j)}{2} \end{aligned} \quad (13)$$

If we add a label to a comment, say j , then this will result in new values for bias and unbiased ratings for all nodes. Thus, the estimated risk will also change. Note that the estimated risk will be different for different labels ($+1$ or -1) for the comment j . Let $\hat{\text{Risk}}^{+(j,l)}$ denote the estimated risk after adding comment j and label l . Again, since we don't know the true label l for j , we use $r(j)$ to estimate the probability of $L(j) = l$ and weigh the risk for each label l by the probability estimates. Therefore, the estimated expected risk after adding comment j will be

$$\hat{\text{Risk}}^{+j} = \frac{(1 + r(j))}{2} \hat{\text{Risk}}^{+(j,1)} + \frac{(1 - r(j))}{2} \hat{\text{Risk}}^{+(j,-1)} \quad (14)$$

We choose the comment k to label that minimizes the estimated expected risk as

$$k = \arg \min_j (\hat{\text{Risk}}^{+j})$$

To pick a comment to label, we need to find the expected risk $\hat{\text{Risk}}^{+j}$ for each comment j which involves computing unbiased rating and bias scores for all nodes in the graph using our semi-supervised algorithm. Moreover, we run the

Algorithm 1 COMPUTENEWRATINGS

Input: User-comment graph G , existing ratings r , Labeled comments L , comment j , new label l for j , neighborhood hops k ;

Output: New ratings r' for comments in $N_k(j)$;

Let G' be the subgraph of G comprising nodes in $N_k(j)$;
Let X be the set of comments j' in G' such that neighbors of j' in G are not all in G' ;

Fix ratings r' of all comments in $L \cup X$ to their rating values in r ;

Fix rating $r'(j)$ to l ;

Run semi-supervised (iterative) algorithm on G' keeping ratings of comments in $L \cup X \cup \{j\}$ fixed;

Let r' be the new ratings for comments j' in G' ;

return $\{r'(j') : j' \in N_k(j)\}$;

algorithm twice to compute $\hat{\text{Risk}}^{+(j,l)}$ for each possible label l (-1 and $+1$). Thus, each iteration of the active learning procedure has time complexity $O(mn)$, and it is very expensive to find a single comment to label. Even if we use a matrix based procedure to update the risks (as in [29]), it is still very expensive with time complexity $O(n^2)$. Also, the matrix based procedure requires the matrix to be inverted initially. This motivates us to design a fast and scalable technique to approximate expected risk. Here, we exploit the fact that the unbiased rating and bias score of a node depends primarily on the nodes in its close vicinity (see Section 6.2).

5.2 Scalable learning algorithm

At the core of our scalable active learning algorithm is an extremely fast procedure for computing new ratings r' (from current ratings r) when a single comment j is assigned a label l . Let $N_k(j)$ denote the comments within k hops of comment j , and let G' be the subgraph of G consisting of nodes in $N_k(j)$. In Section 6.2, we make the following key observation: a node's influence decreases exponentially as the length of the path is increased. As a result, the label assignment to j mainly affects the ratings of comments in the immediate neighborhood of j . Thus, we only need to compute new ratings for comments in $N_k(j)$ (for a small k) which can be accomplished very efficiently by running our semi-supervised iterative algorithm on the much smaller graph G' instead of the entire graph G . And these new ratings for comments in $N_k(j)$ can in turn be used to efficiently approximate the expected risk $\hat{\text{Risk}}^{+(j,l)}$. Note also that each time a comment j is labeled, only the ratings for comments in its neighborhood $N_k(j)$ change, and so we only need to re-estimate the expected risk $\hat{\text{Risk}}^{+(j',l)}$ for comments j' in its $2k$ -hop neighborhood $N_{2k}(j)$. This is because the $\hat{\text{Risk}}^{+(j',l)}$ values of only comments $j' \in N_{2k}(j)$ could have changed since these are the only comments whose k -hop neighborhoods have comments with new (changed) ratings.

Algorithm 1 describes our procedure for computing the new ratings r' of comments in the k -hop neighborhood of a comment j when its label is set to l . As mentioned earlier, this procedure is at the core of our active learning algorithm. Since G is bipartite, parameter k is set to an even value,

Algorithm 2 FASTACTIVELEARNING

Input: User-comment graph G , number of labels N , neighborhood hops k ;

Output: Set of labeled comments L with labels;

$L = \emptyset$;

Run semi-supervised algorithm on G to compute ratings r ;

$\hat{Risk} = \sum_{i=1}^n \frac{1-r^2(i)}{2}$;

$S = \{1, \dots, n\}$;

while $|L| < N$ **do**

for each $j \in S$ **do**

$\{r'(j')\} = \text{COMPUTENEWRATINGS}(G, r, L, j, +1, k)$;

$\hat{Risk}^{+(j,1)} = \hat{Risk} + \sum_{j' \in N_k(j)} \frac{r(j)^2 - r'(j')^2}{2}$;

$\{r'(j')\} = \text{COMPUTENEWRATINGS}(G, r, L, j, -1, k)$;

$\hat{Risk}^{+(j,-1)} = \hat{Risk} + \sum_{j' \in N_k(j)} \frac{r(j)^2 - r'(j')^2}{2}$;

$\hat{Risk}^{+j} = \frac{(1+r(j))}{2} \hat{Risk}^{+(j,1)} + \frac{(1-r(j))}{2} \hat{Risk}^{+(j,-1)}$;

end for

Select comment j with highest expected estimated risk

\hat{Risk}^{+j} and let l be the label for j ;

$L = L \cup \{j\}$;

$r(j) = L(j) = l$;

$\{r'(j')\} = \text{COMPUTENEWRATINGS}(G, r, L, j, l, k)$;

$\hat{Risk} = \hat{Risk} + \sum_{j' \in N_k(j)} \frac{r(j)^2 - r'(j')^2}{2}$;

Set ratings $r(j') = r'(j')$ for comments $j' \in N_k(j)$;

$S = \{j' : j' \in N_{2k}(j)\} - L$;

end while

return $\{L(j) : j \in L\}$;

for e.g., 4. Now, in the subgraph G' comprising nodes in $N_k(j)$, there may be comment nodes j' that are k hops from j such that edge $i \rightarrow j'$ is in G and $i \notin G'$. Since all the neighbors of j' are not contained in G' , our semi-supervised algorithm may not compute its rating precisely, and so we fix j' 's ratings. We also fix the ratings of labeled comments in L and comment j with label l .

Our fast neighborhood-based active learning procedure is described in Algorithm 2. The algorithm keeps track of the total expected risk for all comments in \hat{Risk} . It computes \hat{Risk}^{+j} for a comment j using Equation (14) where $\hat{Risk}^{+(j,l)}$ is estimated from \hat{Risk} and the new ratings r' for comments in $N_k(j)$ returned by COMPUTENEWRATING. Once the comment j with the highest \hat{Risk}^{+j} value is labeled, say with label l , the new ratings r' for comments in j 's neighborhood are re-computed (by invoking COMPUTENEWRATING) and \hat{Risk} is adjusted to reflect the new ratings r' . Also, the \hat{Risk}^{+j} values for comments $j' \in N_{2k}(j)$ whose neighborhoods overlap with j 's neighborhood are re-computed since ratings of comments in j 's neighborhood may have changed. The procedure terminates once N comments have been labeled.

Note that we can periodically re-run our semi-supervised algorithm on the entire graph G to ensure that our ratings and expected risk estimates for nodes remain fairly accurate. To optimize even further, after computing the risk for all the nodes, we can select a batch of the top m comments with the highest risk to label.

6. ANALYTICAL SOLUTION AND CONNECTION WITH OTHER WORK

In this section, we give the analytical solution to our unsupervised method. This can also be generalized to the semi-supervised method.

6.1 Analytical solution

Let A be the adjacency matrix of the graph. Also, let the inverse in-degree matrix be $D_i^{-1} = \text{diag}\{1/d^i(1), \dots, 1/d^i(n)\}$, and likewise, we have the inverse outdegree matrix D_o^{-1} . Assume a zero entry in case $d(i)$ is zero. Let \vec{b} and \vec{r} be the vectors for bias and unbiased rating. The analytical solution of unbiased rating using Equations (5) and (6) is given as follows:

$$\vec{r} = (I - \frac{1}{2}D_i^{-1}A^T D_o^{-1}A)^{-1}(D_i^{-1}A^T - \frac{1}{2}D_i^{-1}A^T D_o^{-1})\vec{1} \quad (15)$$

For completeness, we give a short proof of the existence of $(I - \frac{1}{2}D_i^{-1}A^T D_o^{-1}A)^{-1}$ in Appendix C. We can similarly give an analytical solution to the bias vector \vec{b} as follows:

$$\vec{b} = \frac{1}{2}(I - \frac{1}{2}D_o^{-1}A D_i^{-1}A^T)^{-1}(D_o^{-1} - D_i^{-1}A^T D_i^{-1}A^T D_o^{-1})\vec{1} \quad (16)$$

6.2 Interpretation of the solution

In this subsection, we examine the solution in depth and explore the connection with other works. The first observation we make is that matrices $D_i^{-1}A^T$ and $D_o^{-1}A$ are simply the row-normalized (L_0 or L_1) forms of A^T and A . That is, for any a_{ij} , we divide it with $\sum_j |a_{ij}|$. From here on, we refer to $D_i^{-1}A^T$ and $D_o^{-1}A$ as A'^T and A'' , respectively. Rewriting \vec{r} , we get

$$\vec{r} = (I - \frac{1}{2}A'^T A'')^{-1}\vec{y} \quad (17)$$

Here, $\vec{y} = (D_i^{-1}A^T - \frac{1}{2}D_i^{-1}A^T D_o^{-1})\vec{1}$. The matrix $A'^T A''$ is a co-citation matrix with some normalization. It means that if two users have given a similar score (+1 or -1) to many comments, then their co-citation score would be high. However, if their opinions differ (they give different scores), then the co-citation score would be negative. Note that when we typically refer to a co-citation matrix, we assume that matrix would have all non-negative entries. However, this is not the case with $A'^T A''$. Entries of the principal eigenvectors of a non-negative co-citation matrix are also the motivation behind the famous HITS algorithm. Therefore, our algorithm also resembles HITS closely, especially Randomized HITS [27].

By choosing $(1 - \epsilon)^2 = 1/2$ (see Section 5.1 in [27]), we can notice the close resemblance of Equation (15) with the authority vector \vec{a} of Randomized HITS which is given by

$$\vec{a} = (I - \frac{1}{2}A'_{row} A'_{col})^{-1}\vec{k} \quad (18)$$

Here, vectors \vec{y} (Equation (17)) and \vec{k} (Equation (18)) set prior weights for different nodes. The rating vector \vec{r} is similar to the authority vector. A similar interpretation can be deduced for the bias vector, which relies on the co-reference matrix.

To gain more insight, we now approximate the matrix $(I - \frac{1}{2}A'^T A'')^{-1}$ by a generalization of the geometric series

known as the Neumann series (see Section 4.1 from matrix algorithms [23]), as:

$$(I - \frac{1}{2}A^T A'')^{-1} = I + (\frac{1}{2}A^T A'') + (\frac{1}{2}A^T A'')^2 + \dots \quad (19)$$

Notice that higher-order terms are higher-order co-citation matrices. An element $[(A^T A'')^l]_{ij}$ represents the co-citation score between nodes i and j using network paths of length l . We can observe that if the length of a path is increased by 1, its contribution decreases by 1/2, indicating that bias and unbiased rating scores of nodes are heavily influenced by other nodes in their neighborhood. Here, the first few terms provide a good approximation of the inverse matrix.

Category	# votes	# comments	# thumbs-up	# users
Business (bs)	3.7M	89.8K	2.3M	98K
Science (sc)	871K	21.8K	491K	39K
US (us)	7.2M	156.9K	4.4M	171K
Politics (pl)	1.4M	37.1K	866K	44K
World (wl)	4M	111K	2.3M	112K

Table 1: Dataset description.

7. EXPERIMENTS

In this section, we give experimental evidence of the superiority of our techniques under different settings. We then discuss the topicality of user bias, and finally, we show the convergence rate of the algorithm.

7.1 Experimental Setup

7.1.1 Dataset description

The dataset consists of comments from Yahoo! News. Each comment belongs to a particular news article pre-categorized as politics (pl), business (bs), world (wl), science (sc), and US (us). We use the terms category and topic interchangeably throughout this section. While creating the dataset, we considered the votes of only frequent users (minimum ≈ 10 votes), and extracted comments with at least ≈ 10 votes from frequent users. Therefore, it is possible to have users with less than 10 votes in the dataset. Table 1 contains a description of the dataset for different categories – each category dataset comprises comments belonging to the category and users that rate these comments.

7.1.2 Evaluation methodology

We collected editorial reviews on 456 comments from three editors across all categories. Editors followed general editorial guidelines while rating a comment such as harms minors in any way, content that is unlawful, harmful, threatening, abusive, harassing, tortuous, defamatory, vulgar, etc. We consider the mean rating (between -1 and +1) provided by the three editors as the true label of a comment. We consider this editorial rating as the ground truth, and compare our techniques, namely unsupervised method, semi-supervised method with random labels, and semi-supervised method with active learning. We also give a detailed comparison with the simple mean of user ratings, which is currently used in most systems. We run the algorithms separately for each topic.

As discussed in Section 3.1, other works [13, 17] have different notions of bias which do not fit well in the comment-rating environment. We present the results for a representative approach in Section 7.2.5.

7.1.3 Performance metric

We refer to the rating returned by the unsupervised and semi-supervised techniques as the *unbiased rating*, and the rating based on the mean ($\text{avg}_i\{w_{ij}\}$) as the *mean rating*. We compare these two ratings with the 456 available editorial ratings score (which we assume to be the ground truth), and compute the mean-square-error (MSE). We also present the relative improvement of our algorithms over mean rating. The relative improvement is the percentage decrease in MSE over mean rating.

7.2 Experimental Results

7.2.1 Unsupervised Setting

In this subsection, we compare the results of our unsupervised technique with the mean rating. We show the MSE of these two ratings in Table 2. The second column depicts the MSE between unbiased ratings and the editorial ratings for different categories. We can observe that even an unsupervised approach significantly reduces the error over the mean rating (in the third column).

The percentage decrease (fourth column) shows the relative improvement our algorithm has over the mean rating. In all the categories, we have significantly outperformed mean rating with at least 35% improvement. Observe that our algorithm has much better performance in categories such as science and US. In later experiments, we only show the percentage improvement rather than the actual MSE numbers as it helps in easy comparison across categories.

	Unbiased Rating Error	Mean Rating Error	% decrease
bs	0.21	0.34	38
sc	0.16	0.29	44
us	0.18	0.34	48
pl	0.12	0.2	36
wl	0.22	0.37	39

Table 2: Results for our unsupervised algorithm. The unsupervised technique outperforms the baseline mean rating on all topics.

7.2.2 Semi-supervised Setting

In the unsupervised setting, we ran the algorithm on nearly 400K comments and tested the output on 456 labeled comments. In the semi-supervised algorithm, we supply some labels and then test the algorithm. However, even for semi-supervised learning, we require a meaningful number of labeled comments. But 456 is too small for both training and testing. Therefore, we created a smaller dataset for our semi-supervised algorithm. We built a new dataset with these 456 labeled comments representing 10% of the data. Comments which received the most ratings constitute the rest ($\approx 90\%$). This ensures that the resulting graph is well connected. Note here that connectivity is important because running the algorithm on two disconnected components is the same as running on them separately.

Semi-supervised with random labels: In this setting, we compare the unsupervised and semi-supervised (with random labels) algorithms. In the semi-supervised setting with random labels, we supply roughly 50% of the labels (representing $\approx 5\%$ of the data) to the algorithm. We test on

the remaining 50% of the labels. The relative improvement over the mean rating for different values of α is displayed in Table 3. We can observe that with only 5% labeled comments, the performance of semi-supervised has improved significantly over the unsupervised technique. We obtained the best performance for $\alpha = 10$. For a category such as politics, we indeed see a dramatic improvement.

Another interesting comparison is between the unsupervised algorithm on the larger dataset (Table 2) and the smaller subset (Table 3). Running the algorithm on the bigger dataset generally gave superior performance (except in category business). The implication here is that the algorithm improves with more data.

	Semi-supervised (random labels)			Unsupervised
	$\alpha = 1$	$\alpha = 3$	$\alpha = 10$	
bs	36	37	40	34
sc	26	26	27	26
us	46	47	49	46
pl	59	62	64	30
wl	35	36	38	33

Table 3: Comparison between the unsupervised and semi-supervised algorithm. Semi-supervised shows improvement, especially in category politics.

Semi-supervised with active learning: The goal of the active learning technique is to maximize performance with few labels. Therefore, we let the algorithm pick 10% of the labeled data (that constitutes only 1% of all the comments). Recall that the labeled data comprises 456 comments. We compare our active learning scheme with the earlier semi-supervised algorithm with random labels (50% of the labeled data). The results are shown in Table 4. With just 1% labeled comments (chosen actively), the active learning algorithm outperforms random that uses 5% in most of the categories (except for politics). Here, note that we restrict our active learning method to pick comments only from the available set of editorial ratings, and this may be sub-optimal. In an ideal scenario, however, our algorithm can select any comment to label. Therefore, we expect active learning to perform even better in practice.

	Semi-supervised	
	Active Learning labeled = 1%	Random labels labeled = 5%
bs	37.8	36.4
sc	39.0	26.7
us	48.9	46.6
pl	29.7	59.9
wl	35.7	35.1

Table 4: Comparison between random labels and labels identified with active learning. In general, active learning gives better performance with fewer labels.

7.2.3 Topicality of bias

We examine the user’s bias across five categories. We mentioned earlier that a user could be biased on certain topics. This phenomenon of topicality is discussed in this subsection, as well as in some earlier work [8].

We compiled a list of users who voted on comments across all the topics and thus, we have their bias scores (computed by our unsupervised algorithm). Therefore, we have five bias scores for each user. We compute the minimum and maximum bias scores across five categories for each user. The difference between the minimum and maximum scores is a good indication of the topicality of bias. Table 5 shows the difference in bias for the top percentile of users. Observe that there is a significant difference in bias scores even for the 40th percentile.

Percentile	0.1%	1%	5%	10%	20%	30%	40%
Bias	0.43	0.34	0.27	0.23	0.2	0.17	0.15

Table 5: Maximum bias difference of a user across different categories.

7.2.4 Analysis of rate of convergence

We iterate until the L_1 difference between two consecutive iterations across all the nodes becomes very small. We refer to this difference as error, calculate it as follows:

$$error = \sum_i |f^{t+1}(i) - f^t(i)| \quad (20)$$

Here, function $f^t(i)$ represents the bias/unbiased rating of node i , and at iteration t . As we continue to iterate, we expect the error to decrease.

Earlier, in Theorem 1, we proved that the error between two consecutive iterations decreases exponentially as we perform more iterations. In Figure 2, we show the errors of unbiased rating and bias after each iteration. We can observe that the error has reduced significantly after a few iterations. Theoretically, we expect the error to decrease from 1000 ($\approx 2^{10}$) to 1 after at most 10 iterations. However, we can observe that in practice the decrease is much faster than anticipated ($\approx 10^5$ to 1 in 10 iterations).

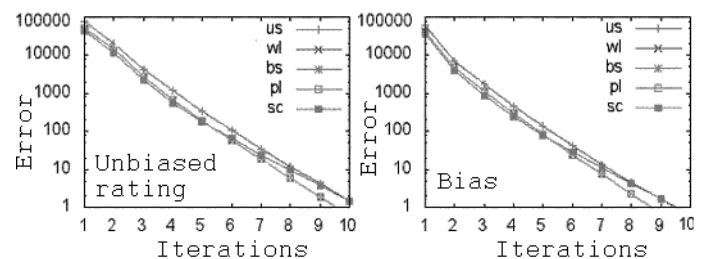


Figure 2: Error decreases exponentially with iterations.

7.2.5 Comparison with Other Schemes

Related approaches, e.g., [17] consider bias to be the propensity of a user to consistently give higher or lower ratings than others. In this subsection, we consider one such representative approach [17] that uses an iterative algorithm to capture the above notion of bias. Figure 3 shows the rating scores obtained by running the algorithm of [17] on the science category after 10 iterations. Unfortunately, the algorithm did not converge (for any topic), returned ratings that are out of

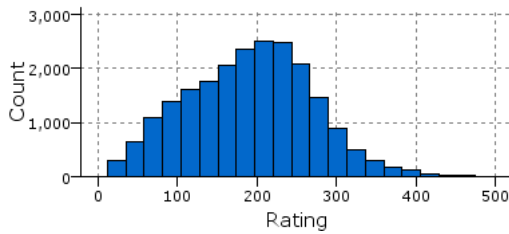


Figure 3: Ratings histogram obtained for the algorithm of [17].

scale (0 to 500), and was very sensitive to the initial choice of seed values. This does not allow us to make a meaningful comparison with our algorithm and the baseline mean rating.

8. RELATED WORK

Comment quality in social media has been studied in great detail (see [8] and references therein). There are different aspects of quality [11, 24] such as relevance, validity, flaming, etc. It is important to hide or delete low quality content. Since the volume of comments on the internet is huge, utilizing the knowledge of crowds appears to be a viable option. Techniques to analyze and improve the quality of content through moderation have been studied in [14, 15].

Going beyond moderation, there have been several research efforts aimed at ranking comments. [10] proposes to rank comments in the absence of explicit positive and negative ratings. It uses a machine learning based approach to train a regression model with features based on content, message visibility, and user reputation. [22] investigates comment usefulness, another related quality measure, on YouTube. It employs bag-of-word features to train classification models for deciding the usefulness of new comments. User votes are not used as features, instead they are used for defining training data, i.e., comments that receive many positive votes are assumed to be useful. Note that this problem is complementary to ours. Because of bias, comments with many positive votes may still be abusive – our bias correction schemes help to address this. In [1], generalizations of matrix factorization that leverage both features and past ratings are used to recommend comments to users.

Author reputation in a comment rating environment is studied in [6]. User bias is defined as the propensity to give excessive thumbs-ups/thumbs-downs, and is factored in while computing reputation. As mentioned earlier in Section 3.1, this definition of bias differs from ours. Similar definitions of bias as the average deviation from the unbiased rating have been considered in [13, 16, 17]. Thus, if a user gives adversarial ratings such as high ratings to poor comments and low ratings to good comments, then it is possible to keep his/her bias artificially close to zero. However, this is not possible with our formulation. Our work is also related to the work on modeling trustworthiness of data [25, 26].

In our work, we require labels for a few comments. Our algorithm combines these labeled and unlabeled comments together to compute user bias and unbiased ratings for unlabeled comments. Such an approach falls under semi-supervised learning where both labeled and unlabeled data are utilized together to come up with better models. Semi-supervised

techniques on graphs [3, 28, 29] usually make the smoothness assumption, i.e., neighboring nodes share similar labels. However, this assumption is not valid in our case.

A comprehensive survey of active learning schemes for selecting data to label can be found in [21]. There has been a limited amount of work on active learning in graphs [5, 9, 29]. To the best of our knowledge, no prior work has considered the bipartite graph similar to our model. Existing results also assume smoothness on graphs. Few heuristics such as [2, 18] also exist, which rely on finding communities and picking a representative. Our technique is based on empirical risk minimization [20, 29], and includes optimizations to incrementally compute risk for neighboring nodes each time a comment is labeled. Thus, our active learning method can scale to large user-comment graphs.

9. CONCLUSIONS

In this paper, we proposed a semi-supervised learning technique for correcting the bias in user ratings. We showed that our algorithm has a number of desirable properties such as convergence, uniqueness, and low computational cost. Furthermore, we presented a scalable active learning algorithm that greedily chooses comments to label such that the overall expected risk is minimized. Empirical results on real-life comments from Yahoo! News show that our semi-supervised method, even without any labeled data, reduces error by as much as 48% over baseline measures. With only 5% random labels, the error reductions increase further to as much as 64%. Furthermore, with active learning we are able to achieve superior performance in most of the categories with only 1% labeled data. An interesting direction for future work is to exploit the textual content of comments along with the structural information used by our algorithms to further improve the accuracy of computed ratings.

10. ACKNOWLEDGMENTS

The authors are grateful to the reviewers for their helpful comments. We thank Deepak Agarwal and Bee-Chung Chen for sharing the dataset, and Vineet Chaoji for the useful feedback.

11. REFERENCES

- [1] D. Agarwal, B.-C. Chen, and B. Pang. Personalized recommendation of user comments via factor models. In *EMNLP*, 2011.
- [2] M. Bilgic, L. Mihalkova, and L. Getoor. Active learning for networked data. In *ICML*, 2010.
- [3] A. Blum and S. Chawla. Learning from labeled and unlabeled data using graph mincuts. In *ICML*, 2001.
- [4] S. Brin and L. Page. The anatomy of a large-scale hypertextual web search engine. *Computer Networks*, 30(1-7):107–117, 1998.
- [5] N. Cesa-Bianchi, C. Gentile, F. Vitale, and G. Zappella. Active learning on trees and graphs. In *COLT*, 2010.
- [6] B.-C. Chen, J. Guo, B. Tseng, and J. Yang. User reputation in a comment rating environment. In *KDD*, 2011.
- [7] N. Diakopoulos and M. Naaman. Topicality, time and sentiment in online news comments. In *CHI EA '11*, 2011.
- [8] N. Diakopoulos and M. Naaman. Towards quality discourse in online news comments. In *CSCW*, 2011.
- [9] A. Guillory and J. Bilmes. Label Selection on Graphs. In *NIPS*. 2009.
- [10] C.-F. Hsu, E. Khabiri, and J. Caverlee. Ranking comments on the social web. In *ICSE*, 2009.

- [11] S. Kiesler, J. Siegel, and W. McGuire, Timothy. Computer-supported cooperative work: a book of readings. chapter Social psychological aspects of computer-mediated communication (Reprint), pages 657–682. Morgan Kaufmann Publishers Inc., 1988.
- [12] J. M. Kleinberg. Authoritative sources in a hyperlinked environment. *J. ACM*, 46(5):604–632, 1999.
- [13] Y. Koren. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *KDD*, 2008.
- [14] C. Lampe and P. Resnick. Slash(dot) and burn: distributed moderation in a large online conversation space. In *CHI*, 2004.
- [15] C. A. Lampe, E. Johnston, and P. Resnick. Follow the reader: filtering comments on slashdot. In *CHI*, 2007.
- [16] H. W. Lauw, E.-P. Lim, and K. Wang. Bias and controversy: beyond the statistical deviation. In *KDD*, 2006.
- [17] H. W. Lauw, E.-P. Lim, and K. Wang. Summarizing review scores of “unequal” reviewers. In *SDM*, 2007.
- [18] S. A. Macskassy. Using graph-based metrics with empirical risk minimization to speed up active learning on networked data. In *KDD*, 2009.
- [19] K. Purcell, L. Purcell, A. Mitchell, T. Rosenstiel, and K. Olmstead. Understanding the participatory news consumer. *Pew Internet and American Life Project*, 2010.
- [20] N. Roy and A. McCallum. Toward optimal active learning through sampling estimation of error reduction. In *ICML*, 2001.
- [21] B. Settles. Active learning literature survey. Computer Sciences Technical Report 1648, University of Wisconsin–Madison, 2009.
- [22] S. Siersdorfer, S. Chelaru, W. Nejdl, and J. S. Pedro. How useful are your comments?: analyzing and predicting youtube comments and comment ratings. In *WWW*, 2010.
- [23] G. W. Stewart. *Matrix Algorithms: Volume 1, Basic Decompositions*. Society for Industrial Mathematics, 1998.
- [24] D. M. Strong, Y. W. Lee, and R. Y. Wang. Data quality in context. *Commun. ACM*, 40:103–110, May 1997.
- [25] V. G. V. Vydiswaran, C. Zhai, and D. Roth. Content-driven trust propagation framework. In *KDD*, 2011.
- [26] X. Yin, J. Han, and P. S. Yu. Truth discovery with multiple conflicting information providers on the web. *IEEE Trans. Knowl. Data Eng.*, 20(6):796–808, 2008.
- [27] A. X. Zheng, A. Y. Ng, and M. I. Jordan. Stable algorithms for link analysis. In *SIGIR*, 2001.
- [28] X. Zhu, Z. Ghahramani, and J. D. Lafferty. Semi-supervised learning using gaussian fields and harmonic functions. In *ICML*, 2003.
- [29] X. Zhu, J. Lafferty, and Z. Ghahramani. Combining active learning and semi-supervised learning using gaussian fields and harmonic functions. In *ICML 2003 workshop on The Continuum from Labeled to Unlabeled Data in Machine Learning and Data Mining*, pages 58–65, 2003.

APPENDIX

A. PROOF OF LEMMA 1

LEMMA 1. *The difference of ratings in two ratings, p and q is at most 1.*

$$|r^p(i) - r^q(i)| \leq 1$$

PROOF. If $i \in L$, then $|r^p(i) - r^q(i)| = 0$. We now prove where comment i is not labeled.

$$\begin{aligned} |r^p(i) - r^q(i)| &= \left| \frac{1}{d^i(j)} \sum_{i:i \rightarrow j} w_{ij} (bias^{q-1}(i) - bias^{p-1}(i)) \right| \\ &\leq \frac{1}{d^i(j)} \sum_{i:i \rightarrow j} |bias^{q-1}(i) - bias^{p-1}(i)| \leq \frac{1}{d^i(j)} \sum_{i:i \rightarrow j} 1 = 1 \end{aligned}$$

Last inequality comes from the fact that $bias^*(i) \in [0, 1]$, therefore $|bias^*(i) - bias^*(i)| \leq 1$. \square

B. PROOF OF THEOREM 1

We show the bound of Theorem 1 of Section 4.2.1. We prove this using mathematical induction to prove the error bound.

PROOF. **Basis:** We first prove for $t = 1$.

$$\begin{aligned} &|bias^2(i) - bias^1(i)| \\ &= \left| \frac{1}{2 \cdot (\alpha \cdot d_L^o(i) + d_{UL}^o(i))} \left(\sum_{j:i \rightarrow j, j \in UL} w_{ij} (r^1(j) - r^0(j)) \right) \right| \\ &\leq \left| \frac{1}{2 \cdot (\alpha \cdot d_L^o(i) + d_{UL}^o(i))} \left(\sum_{j:i \rightarrow j, j \in UL} |r^1(j) - r^0(j)| \right) \right| \\ &\leq \left| \frac{1}{2 \cdot (\alpha \cdot d_L^o(i) + d_{UL}^o(i))} \left(\sum_{j:i \rightarrow j, j \in UL} 1 \right) \right| \text{ [using Lemma 1]} \\ &= \frac{d_{UL}^o(i)}{2 \cdot (\alpha \cdot d_L^o(i) + d_{UL}^o(i))} \leq \frac{1}{2} \end{aligned}$$

This confirms the basis.

Induction step: We assume the bound to be true for $b^t(i)$, i.e., for the t^{th} iteration. In the $(t+1)^{\text{th}}$ iteration,

$$\begin{aligned} &|bias^{t+2}(i) - b^{t+1}(i)| = \\ &= \left| \frac{1}{2 \cdot (\alpha \cdot d_L^o(i) + d_{UL}^o(i))} \left(\sum_{j:i \rightarrow j, j \in UL} w_{ij} (r^{t+1}(j) - r^t(j)) \right) \right| \\ &\leq \left| \frac{1}{2 \cdot (\alpha \cdot d_L^o(i) + d_{UL}^o(i))} \left(\sum_{j:i \rightarrow j, j \in UL} \left(\frac{1}{d^i(j)} \sum_{k:k \rightarrow j} |w_{kj}| |bias^{t+1}(k) - bias^t(k)| \right) \right) \right| \\ &\leq \left| \frac{1}{2 \cdot (\alpha \cdot d_L^o(i) + d_{UL}^o(i))} \left(\sum_{j:i \rightarrow j, j \in UL} \left(\frac{1}{d^i(j)} \sum_{k:k \rightarrow j} \left(\frac{1}{2} \right)^t \right) \right) \right| \\ &\quad \text{[Induction Assumption]} \\ &= \frac{d_{UL}^o(i)}{(\alpha \cdot d_L^o(i) + d_{UL}^o(i))} \cdot \left(\frac{1}{2} \right)^{t+1} \leq \left(\frac{1}{2} \right)^{t+1} \end{aligned}$$

C. PROOF OF EXISTENCE OF THE INVERSE

In this section, we present a small proof of existence of the inverse of matrix $(I - \frac{1}{2} D_i^{-1} A^T D_o^{-1} A)$ (see Section 6).

For a matrix A , if $\rho(A) < 1$, then $(I - A)^{-1}$ exists (see Theorem 4.20 from matrix algorithms [23]), where $\rho(A)$ denotes the spectral radius of A . Let $\|\cdot\|_\infty$ denote the infinity norm; it satisfies $\|A\|_\infty \geq \rho(A)$ and is submultiplicative, i.e., $\|AB\| \leq \|A\| \|B\|$. If we can show that $\|\frac{1}{2} D_i^{-1} A^T D_o^{-1} A\| < 1$, then it will prove $\rho(\frac{1}{2} D_i^{-1} A^T D_o^{-1} A) < 1$ and subsequently, the existence of the inverse.

Now, $\frac{1}{2} \|D_i^{-1} A^T D_o^{-1} A\|_\infty \leq \frac{1}{2} \|D_i^{-1} A^T\|_\infty \|D_o^{-1} A\|_\infty$. Observe that the maximum absolute sum of each row of $D_i^{-1} A^T$ and $D_o^{-1} A$ is 1, which shows that $\|\cdot\|_\infty$ of these two matrices is at most 1. Therefore, $\rho(\frac{1}{2} D_i^{-1} A^T D_o^{-1} A) \leq \frac{1}{2}$. \square