WorkerRank: Using Employer Implicit Judgements To Infer Worker Reputation

Technical Report UCSC-SOE-14-01, School of Engineering, UC Santa Cruz, 2014

Maria Daltayanni
UC Santa Cruz
mariadal@cs.ucsc.edu

Luca de Alfaro
UC Santa Cruz
luca@cs.ucsc.edu

ABSTRACT

In online labor marketplaces two parties are involved; employers and workers. An employer posts a job in the marketplace to receive applications from interested workers. After evaluating the match to the job, the employer hires one (or more workers) to accomplish the job via an online contract. At the end of the contract, the employer can provide his worker with some rating that becomes visible in the worker online profile. This form of explicit feedback operates as a recommender to guide future hiring decisions, since it is indicative of worker true ability. In this paper, first we discuss some of the shortcomings of the existing reputation systems that are based on the end-of-contract ratings. Then we propose a new reputation mechanism that uses Bayesian updates to combine employer implicit feedback signals in a link-analysis approach. The new system addresses the shortcomings of existing approaches, while yielding better signal for the worker quality towards hiring decision.

Categories and Subject Descriptors

H.4.4.3.4 [**Information Systems**]: World Wide Web—*Web applications, Crowdsourcing, Reputation Systems*

General Terms

Algorithms, Design, Experimentation, Performance

Keywords

Elo Ratings, Link Analysis, Reputation Systems, Crowdsourcing

1. INTRODUCTION

In online labor marketplaces, such as oDesk [1], Elance [2] and Freelancer [3], two parties are involved; employers and workers. Employers post job openings and candidate workers apply to them, based on their qualifications, skills and interests. The employers review the applicants' online resumes, and interview few applicants to take hiring decisions. The worker reputation, i.e., the ratings that the worker has received in his past jobs in the platform, is one of the most important considerations for the employer hiring decision, since it reveals how other employers evaluate the worker true ability in real job scenarios. Although the reputation information is a useful signal, reputation scores are usually *skewed* towards high ratings [17], because employers care about the impact of their feedbacks on the workers' future opportunities for jobs in the marketplace. The skewed distribution of ratings makes them less helpful in identifying very competent workers.

The reputation signal is also very *sparse*, since a worker needs to apply, get hired and complete few jobs to obtain a representative reputation score. In common recommender systems, an unknown rating implies that we have no explicit information about the employer's preference for the worker. In that case we need to build a model to predict the unknown information, or, alternatively make inferences from the employer's behavior[1].

To address the limitations of the existing reputation systems in labor marketplaces, we present *WorkerRank*, a new reputation system that leverages employers' implicit judgements at the application evaluation moment, rather than the employer's explicit feedback at the job completion moment. Although the implicit judgments are more noisy than the explicit ones, they are more broadly available, since the number of applications is usually one to two orders of magnitudes higher than the number of hires. Moreover, the implicit actions of the employers are not revealed and, consequently, the employers do not bias their judgments towards high ratings (as happens when they aim to avoid the negative impact on the workers). As a result, the obtained ratings are not skewed.

We consider an employer decision to hire worker A, thus ranking A above some other candidate B, as an input that "A won over B" in a match. The employer decisions can thus be interpreted as a set of match outcomes. There are many algorithms ([14], [15], [16], [29]) that can be used to aggregate match outcomes. Our reputation system builds upon the Elo ratings system[14] that is widely used to evaluate chess players. In particular, we assign each worker an initial rating and we treat the applicants to a job opening as the participants in a chess tournament. Applicants that get hired get their scores increased and those who are rejected get their scores decreased. The extent of the increase or the decrease depends upon the ratings of the other applicants, i.e., the better the rejected applicants are, the more the rating of the hired contactor increases. Similarly, the worse the hired applicants are, the more the ratings of the rejected applicants decrease.

To deal with the noise of implicit judgments, we assign each em-

[1] http://www.odesk.com
[2] http://www.elance.com
[3] http://www.freelancer.com

ployer a score that quantifies the agreement of his decisions with the observed quality of the workers. We then use the obtained scores to weigh the employer judgments. For example, if an employer tends to take decisions that are very different from the rest of the employers, his score will be low and his hiring decisions will have a small impact on the worker ratings. The rest of the paper is organized as follows. In Section 2 we present some notation and in Section 3 we introduce WorkerRank, the new proposed reputation system. We evaluate the new reputation approach on a real-world dataset from oDesk in Section 4. Our results show that the new reputation system not only provides information for far more workers in the marketplace, but it also serves as a better discriminatory signal for hiring decisions. In Section 5 we dicuss some related work and we conclude in Section 6.

2. NOTATION

We represent the labor marketplace data with a directed bipartite graph $G = (U, V, A)$ (Figure 1); U is the set of jobs posted by employers within a specific time period; V is the set of workers who applied to the posted jobs (see Figure 1). Edge $(v, u) \in A$ represents the application of worker $v \in V$ to job $u \in U$. Edge $(u, v) \in A$ represents the employer action on the the worker's application. We consider the following six employer actions:

- *hire*, the employer hires the worker;
- *interview*, the employer contacts the worker to obtain a better understanding of his skills, but the worker is not eventually hired;
- *shortlist*, the employer shortlists the worker for future consideration, but the worker is not invited for interview;
- *ignore*, the employer reviews the worker online resume, but he takes no action on it;
- *hide*, the employer reviews the worker resume and he "hides" the applicant without notifying him; and
- *reject*, the employer reviews the worker resume and notifies him that he will not be considerd for the job.

Among the six actions, we consider the first three as positive indications of the worker ability to accomplish the posted job, while the last three are rather negative. We also assume that the employer actions imply some ranking on the applicant perceived ability to accomplish the job in the following decreasing order: hire > interview > shortlist > ignore > hide > reject. For example, a worker that is selected to be interviewed is considered as a better fit for the job than a worker who is ignored.

The goal of this paper is to compute a score $r(v)$ for each worker v that is informative of the worker ability to accomplish the jobs that he applies to. A score $r(v)$ is considered informative if the relative difference between scores $r(v)$ and $r(v')$ for workers v and v' is predictive for the relative ranking of v and v' in the future jobs that they apply.

3. PROPOSED REPUTATION SYSTEM

In this section we describe a reputation system that builds upon the employer decisions on the worker applications. In Section 3.1 we provide our generic approach and in Section 3.2 we show how we can improve our scores by leveraging job specific information. Finally, in Section 3.3 we discuss how we can combine our reputation system scores with the end-of-contract ratings to obtain a hybrid reputation system.

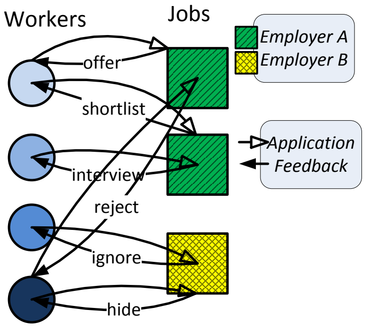

Figure 1: Bipartite graph between workers and jobs posted by employers

3.1 WorkerRank

The WorkerRank reputation system assigns *reputation* scores $r(v)$ to each worker $v \in V$. Along with reputation scores WorkerRank also computes an *importance* score $b(u)$ for each opening $u \in U$ that reflects how a job is important in terms of how objective its employer is when judging candidates. The scores are computed via an iterative calculation process on the application graph G as depicted in Algorithm 1.

Algorithm: In step 1 we initialize reputation $r(v)$ of each worker v to 1.0 and importance $b(u)$ of each opening to 1.0. Then, in steps 4 - 13 we update the worker scores by considering every pair (v, v') of worker applications at a job u to be a tournament with possible outcome of matches:

$$t(v, v', u) = \begin{cases} 0, & \text{if } v \text{ lost against } v' \text{ at job } u \\ 0.5, & \text{if } v \text{ came to draw with } v' \text{ at job } u \\ 1, & \text{if } v \text{ won against } v' \text{ at job } u \end{cases} \quad (1)$$

At step 4, we compute $T_{v,u}$ as the sum of the actual points that v scored in job u against the other opponent candidates. At step 5, we compute $X^i_{v,u}$ as the sum of expected points that v would earn at iteration i against each opponent candidate $v' \neq v$ at job u, according to Elo's formula[14]:

$$t^i_{elo}(v, v', u) = \frac{1}{1 + 10^{(r^i(v') - r^i(v))/400}}, \forall v' : (v', u) \in A \quad (2)$$

For example, consider workers v_1, v_2, v_3 and v_4 who applied to a job; v_1 gets an offer, v_2 is interviewed but never hired and v_3 and v_4 are rejected without interview. Each applicant participates in 3 games versus the other applicants. Worker v_1 wins all three games versus v_2, v_3 and v_4, since he received an offer which is the most positive employer judgement. Worker v_2 loses against v_1 but wins over v_3 and v_4. Finally, each of the workers v_3 and v_4 loses in the games against v_1 and v_2 but they draw when they face each other. The worker points in this job are 3 for v_1, 2 for v_2 and 0.5 for either of v_3 and v_4.

At step 9, the rating update $\delta(v, u)$ of worker v due to his application to job u in the i-th iteration is calculated as follows:

$$\delta^i(v, u) = b^{i-1}(u)(T_{v,u} - X^{i-1}_{v,u}) \quad (3)$$

where $b^{i-1}(u)$ is the importance score of job u from the $i - 1$-th iteration, $T_{v,u}$ is the sum of the actual points that v scored in job u and $X^{i-1}_{v,u}$ is the sum of points he was expected to score based on his rating and the ratings of the other applicants from the previous iteration. Note that the more the applicants in an opening, the

Algorithm 1 WorkerRank: Compute Workers Reputation Scores and Jobs Importance Scores

```
Input: Graph $G = (U, V, A)$
Output: Reputation scores $r(v)$ for workers $v \in V$, importance
    scores $b(u)$ for jobs $u \in U$
 1: Initialize $i = 0$; $r^0(v) = 1, \forall v \in V$; $b^0(u) = 1, \forall u \in U$
 2: repeat
 3:     for $(v, u) \in A$ do
 4:         $T_{v,u} \leftarrow \sum_{v':(v',u)\in A, v'\neq v} t(v, v', u)$
 5:         $X^i_{v,u} \leftarrow \sum_{v':(v',u)\in A, v'\neq v} t^i_{elo}(v, v', u)$
 6:     end for
 7:     $i \leftarrow i + 1$
 8:     for $(v, u) \in A$ do
 9:         $\delta^i(v, u) \leftarrow b^{i-1}(u) \cdot (T_{v,u} - X^{i-1}_{v,u})$
10:     end for
11:     for $v \in V$ and $u \in U$ do
12:         $r^i(v) \leftarrow r^{i-1}(v) + \sum_{u:(v,u)\in A} \delta^i(v, u)$
13:         $b^i(u) \leftarrow \frac{n^i_c(e) - n^i_w(e)}{n^i_c(e) + n^i_w(e)}$
14:     end for
15: until convergence of $\delta$
```

more the expected points, since the equation contains a summation term for each applicant. What is more, the higher the difference between the rating of worker v and the ratings of the other applicants of job u, the more the points that v is expected to score. This is particularly useful since application success of an applicant is not independent from the application success of the remaining candidates at a particular job.

Finally, to obtain the rating of worker v at the i-th iteration, at step 12 we add the average of his partial rating updates (Equation 3) to his rating from the previous iteration $r^{(i-1)}(v)$:

$$r^i(v) = r^{i-1}(v) + \sum_{u:(v,u)\in A} \delta^i(v, u) \tag{4}$$

After calculating the worker ratings, at step 13 we compute the importance scores of jobs $b(u)$. The intuition in Formula 5 is to give more credence to rating updates (Equation 4) that come from jobs posted by unbiased employers. Instead of calculating an independent score for each job, we assign the same score to all jobs that come from the same employer. The importance score is high for employers who make decisions that respect the worker ratings and it is low for employers who do not. Notice that in the end of the algorithm iterations, the worker ratings is an outcome taken out of the aggregation of all employers judgements (step 12). What is more, Elo scoring is based on a self-correcting rating system (convergence at step 15 depends on estimation accuracy). Hence the following formula reflects importance of a job as a measure of judgment deviation between the job's respective employer and the rest employers:

$$b^i(u) = b(e) = \frac{n^i_c(e) - n^i_w(e)}{n^i_c(e) + n^i_w(e)} \tag{5}$$

$e \in E$ denotes employer (in the set of employers E) who posted job u, $n_c(e)$ denotes the number of applicant pairs that were "correctly" ranked by employer e in all of the jobs that he posted and $n_w(e)$ denotes the number of applicant pairs that were "erroneously" ranked. We regard a pair of applicants as correctly ranked if the employer prioritizes the applicant with the highest reputation score. For example, if applicants v and v' have ratings $r(v) = 1$ and $r(v') = 2$ and employer e hires v' and rejects v, then the pair is considered to be correctly ranked.

3.2 Skill WorkerRank

The approach described in Section 3.1 predicts a reputation score for each worker based on application data. That score is computed in a global scope over all jobs where workers have applied. Although global scores provide a signal for the quality of workers, they may not be as powerful to discriminate among similar score workers and guide hiring with accuracy. For example, consider candidates v_1, v_2 in the example shown in figure 2. Also assume that v_2 is better in java than v_1, however v_1 is better overall than v_2 which can be caused from the fact that v_1 has applied to more jobs where he has been successful (offer). At this point our reputation system would prioritize v_1 in the candidates recommendation list, missing the fact that v_2 is better in java.

Our goal is to achieve higher accuracy at predicting worker quality and recommending candidates appropriate for the particular job. As mentioned above, besides the information regarding which worker applied to which job, there is further information regarding skills; what skills each job requires and what skills each worker claims in their profile description. Given this information, we use WorkerRank to derive scores for candidates in a skill-wise fashion, such that eventually we learn how good each worker is at each particular skill. Then, if a job requires a skill, we may rank candidates according to their reputation scores at that particular skill and suggest the top ranked for getting hired. In the used example, workers v_1, v_2 who both claim to be experts in java (and they apply to job u_2 which requires java) will obtain a java score that will show their quality in that skill. The expectation is that recommending v_2 will lead to successful hiring decision.

An algorithm for skill-wise reputation: We consider set of skills S, where $S_u \subseteq S$ denotes the skills required for job $u \in U$ and $S_v \subseteq S$ denotes the skills claimed by worker $v \in V$. Also, we consider bipartite graph $G_S = (U \times S, V \times S, A_S)$ similar to the definition in Section 3.1, where:

- each worker node v is replaced by set of pair $\{$worker, skill$\}$ nodes, $\{v, s\}$, one node for each skill claimed by the worker
- each job node u is replaced by set of pair $\{$job, skill$\}$ nodes, $\{u, s\}$, one node for each skill required for the job
- each (worker, job) edge (v, u) is replaced by set of pair ($\{$worker, skill$\}$, $\{$job, skill$\}$) edges, $(\{v, s\}, \{u, s\})$, one edge for each skill that the worker claims and the job requires.

Then we run Algorithm 1 on G_S. The reputation and importance scores are derived skill-wise, such that we obtain a set of reputation scores $r(v, s),$ where$(v, s) \in V \times S$ and a set of importance scores $b(u, s),$ where$(u, s) \in U \times S$. Obtaining reputation scores for each $\{$worker, skill$\}$ pair provides information about the performance of the worker at the particular skill.

Figure 2 describes an example for graphs G and G_S, with candidates v_1, v_2, v_3 applying to jobs u_1, u_2. Workers v_1, v_3 apply to job u_1 (v_1 receives an offer) and v_1, v_2 apply to job u_2 (v_2 receives an offer). The skills required for job u_1 are $\{$python, django$\}$ and the skills required for job u_2 are $\{$java$\}$. Worker v_1 claims to have skills $\{$python, java django$\}$, v_2 claims $\{$java$\}$, and worker v_3 claims $\{$python, django$\}$.

Learning the correlation between skills and hires: Looking at the application data (enriched with skills information) we observe that in most cases jobs require more than one skills. In that case, we need to decide a ranking for candidates based on the intersection of their scores on a set of different skills. That ranking will

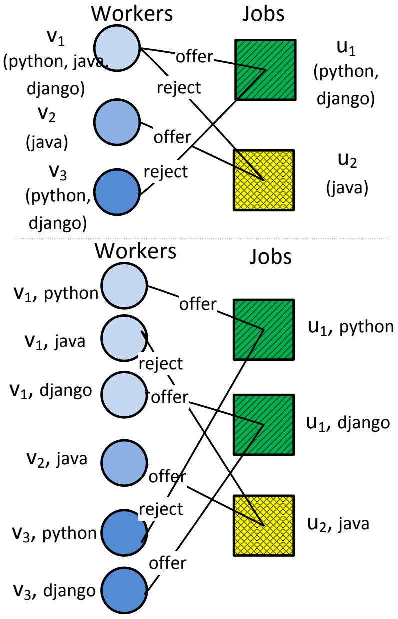

Figure 2: Skill-wise bipartite graph

reflect their suitability for the multi-skill requiring job. In our example where job u_1 requires python and django, we need to rank candidates according to their quality in python and their quality in django. However, the python-score may be more informative about hiring than the django-score. For example it may more beneficial to hire a candidate with a high python-score than hire one with a high django-score. Hence it is important to measure how each skill contibutes towards hiring, before we rank candidates according to skills.

In order to combine a set of scores for each worker across the set of skills required for a job, we allow for a weighted average over the worker's skill-wise scores. We use logistic regression to compute coefficients for skills as features, where we use the binary outcome of the application (hire/no-hire) as the response variable. Coefficients for skill scores will eventually show how informative each skill is about the quality of the worker, measured by the worker's potential of getting hired.

In Algorithm 2 we aggregate skill-wise scores into a single reputation/importance score for each worker/job respectively. The input of the new algorithm is the set of scores derived by Algorithm 1 for the sets of {worker, skill}, {job, skill} pairs. The output of Algorithm 2 is the sets of final reputation scores for each worker and importance scores for each job, after examining his quality across the set of his skills and tuning his overall quality according to the significance of each skill.

In step 1 we consider the set of features F; we consider one's reputation score $r(\cdot, s)$ at a particular skill $s \in S$ to be a feature. We use $f(s) = r(\cdot, s)$ to denote feature regarding reputation score at skill s. For example, if s = python then any (denoted by $\cdot$) worker's score in python, $r(\cdot, \text{python})$, is a feature. Recall that the coefficients we aim to compute, pertain to how each skill score of a worker contributes towards his getting hired. In step 2 we consider the response variable y as the binary hiring outcome $y \in \{hire, no\text{-}hire\}$. Then in step 3 we run logistic regression (LR) on the set of features F with response variable y and we obtain

Algorithm 2 Combine Skill-wise Scores into Reputation

```
Input: Set of skill-wise scores $r(v, s)$, $b(u, s)$, where $(v, s) \in V \times S$, $(u, s) \in U \times S$
Output: Reputation scores $r(v)$ for workers $v \in V$, Importance scores $b(u)$ for jobs $u \in U$
1: Consider feature variables $f(s) \leftarrow r(\cdot, s)$, $\forall s \in S$, and set of feature variables $F \leftarrow \cup_{s \in S} f(s)$
2: Consider response variable $y \leftarrow$ hiring outcome, where $y \in \{hire, no\text{-}hire\}$
3: Learn coefficients $w(f) \leftarrow LR(F, R), \forall f \in F$
4: for $v \in V$ and $u \in U$ do
5:     $r(v) \leftarrow \frac{\sum_{s \in S} r(v, s) \cdot w(s)}{\sum_{s \in S} w(s)}$
6:     $b(u) \leftarrow \frac{\sum_{s \in S} b(u, s) \cdot w(s)}{\sum_{s \in S} w(s)}$
7: end for
```

coefficients $w(f)$ for each skill-score variable $f \in F$. Finally, in steps 5 - 6 we aggregate the input skill-wise scores into using weights across skills learned at step 3. The ouput of the algorithm is single reputation scores for workers (step 5) and single importance scores for jobs (step 6) of the application data.

3.3 Hybrid Model

While it is interesting to compare implicit judgments against explicit judgments to infer a quality measurement for workers, we expect that a hybrid model which combines both, would yield better reults. In this section we use rank aggregation to combine WorkerRank ranking with feedback ranking into an optimal listing of workers such that we predict true ranking (as specified by employer judgments) with higher precision.

In particular, we use the weighted rank aggregation method described in [27]. In this approach the function performs rank aggregation via a Cross-Entropy Monte Carlo algorithm. The algorithm searches for a desired list which is as close to the provided ordered lists as possible. In our implementation we use the Spearman distance to measure the correlation of the ordered lists of implicit and explicit reputation rankinhgs. The convergence criteria used by both algorithms is the repetition of the same minimum value of the objective function in convIn consecutive iterations.

4. EXPERIMENTAL RESULTS

To evaluate the proposed reputation system, we compare WorkerRank with baseline schemes in terms of the sparsity of the signal in the marketplace, the time needed to obtain a signal for new workers, and discriminatory power for hiring decisions. During the evaluation, WorkerRank is compared against a baseline approach which uses data of implicit reputation to rank workers (as a reminder, WorkerRank also runs on *implicit* reputation data). In addition, we compare WorkerRank against ranking approaches that are based on *explicit* reputation data, such as the employers judgments about the performance of workers in accomplished jobs.

4.1 Dataset

We use a sample of real-world application data, along with explicit reputation scores provided by oDesk[25]. The oDesk dataset spans the time period of 53 weeks between March 2012 through March 2013 and it contains approximately 10M applications submitted by 0.5M workers to 1.1M job openings posted by 0.2M employers. In table 1 we provide some statistics regarding the dataset. Tables 2 and 3 show a real job posting example. This example

Table 1: Dataset Statistics

	Training	Testing
Application dates	03/05/12 – 03/10/13	03/11/13 – 03/17/13
#Jobs	1, 151, 859	1, 446
#Workers	477, 464	21, 642
#Employers	232, 014	1, 405
#Applications	9, 214, 557	34, 054
Avg # cands / job	36	41
Min # cands / job	10	10
Max # cands / job	50	50
Median # cands / job	19	23

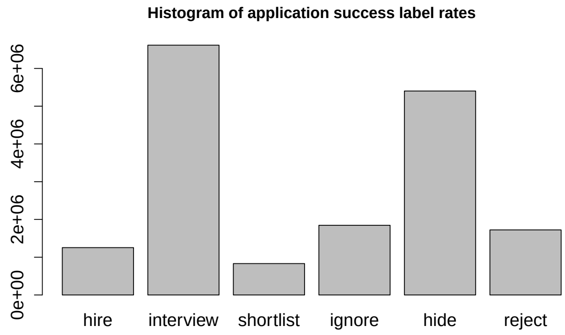

Figure 3: Histogram of application success label rates

includes information about the job title, category, description, required skills, candidate applicants along with their declared skills, hire decision, reputation scores learned via WorkerRank and via skill-based WorkerRank. We observe overlapping skills among the job required skills and the skill-sets declared by the candidates. In Figure 4(b) we show the distribution of the ratings data. As studied in [17], we encounter skewness towards the high rating values.

A baseline for representing the quality of workers based on explicit data, is to collect the ratings assigned to each worker by the employers according to their performance on accomplished jobs, and aggregate them in an average ratings score. Then we estimate the workers quality according to the average employers judgments.

A baseline for representing the quality of workers based on implicit reputation data, that is, based on their activity on past job applications (offer, interview, shortlist, ignore, hide, reject), is to compute their hire rate in the training set. In this approach we collect the sets of jobs at which workers received an offer within the study training period. Then we count the total number of offers for each worker in the training set and we estimate their quality score according to their hire rate.

In the testing phase we rank candidates by hire rate or by explicit ratings reputation to recommend the top ranked for the new job openings.

4.2 Coverage

First, we show that since WorkerRank's results become available at the time of application, the coverage of workers for whom we obtain reputation signal is higher compared to the coverage obtained from explicit reputation. In particular, we run WorkerRank over the applications of the first 52 weeks of our dataset. During this time period we also keep track of the feedback ratings that the workers receive after the end of accomplished jobs. Then we report the number of applications of the 53th week for which there is a WorkerRank score versus the applications for which there is an employer feedback score. Our results show that out of 88, 294 applications in the 53th week, we have WorkerRank scores for 79, 083 (89.6%), while we have feedback scores only for 52, 471 (59.4%). The increase in the marketplace application coverage is 50.8%. We present these results in table 5. Note that the above measurements account for both active and inactive applications.

Table 2: Job Posting Example (The corresponding applicants are shown in table 3)

Title	*Wordpress Developer*
Category	*Web Development*
Required Skills	*wordpress, css, php*
Description	*1.Need Wordpress theme developed. Slider, logo, left sidebar menu, copyright.* *2.Must be experienced Wordpress developer.* *3.Must know CSS, php, Wordpress, html.* *4.Will provide visual design guide.*

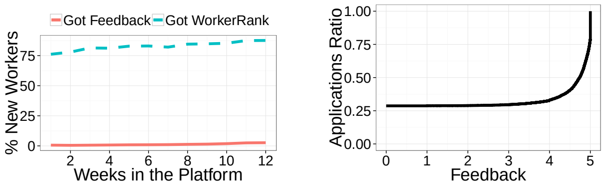

Figure 4: (a)Cold Start, (b)Data Skewness

4.3 Cold Start

Second, we show that WorkerRank is faster in acquiring signal for new workers joining the system, compared to the explicit ratings approach. Since the online marketplaces grow fast, the identification of new competent workers is very significant for their healthy development. For all workers who joined the oDesk platform during the last 12 weeks of our study period, we calculate the percentage of workers for whom we obtain reputation signals within X weeks. X is varying from 1 to 12 weeks. As presented in Figure 4(a), the WorkerRank scores are available for more than 75% of the new workers within one week of their joining the platform and the percetentage ratio grows to 95% after 12 weeks. On the contrary, there are less than 1% of new workers who received feedback at the end of their first week at platform and this percentage does not exceed 5% at the end of the 12-week period.

4.4 Ranking Precision

Third, we show that WorkerRank outperforms baseline approaches in ranking workers by quality. Hence it produces a more reliable system for recommending candidates at new job openings.

4.4.1 *WorkerRank vs Hire Rate vs Explicit Reputation*

MAP

We compare the baseline approaches to WorkerRank using Mean Average Precision (MAP). As shown in Figure 5, with WorkerRank the employer encounters 1 good worker for every 3 workers that he examines in the recommended list. That performance is compared against 1 good worker for every 5 (or, 4 respectively) workers that the employer would encounter if he used the baseline by-hire-rate

Table 3: Skill-based reputation versus global reputation scores

Candidate	App.Success	Skill 1	Skill 2	Skill 3	Skill 4	Skill5	reputation	skill-reputation
v_1	offer	css3	php	wordpress	html	css	2.54	2.28
v_2	**offer**	css3	php	wordpress	html5	css	1.42	**4.67**
v_3	offer	web dev	sofw dev	–	–	–	1.39	2.05
v_4	**reject**	css3	php	wordpress	html5	ajax	**4.82**	3.40
v_5	reject	gr.design	visual-c++	wordpress	web design	illustration	1.38	1.06
v_6	reject	css3	php	javascript	html	jquery	1.05	2.45

Table 4: Performance Improvement: WorkerRank vs Explicit Reputation

	Explicit Reputation	Implicit Reputation		% Improvement	
	Ratings Based (RB)	By Hire Rate (HR)	Workerrank (WR)	WR vs HR	WR vs RB
MAP	0.24	0.21	0.29	**+38.1%**	**+20.8%**
AUC	0.54	0.50	0.61	**+22.0%**	**+13.0%**
% Covered applications	59.4%	–	80.8%	–	**+50.8%**

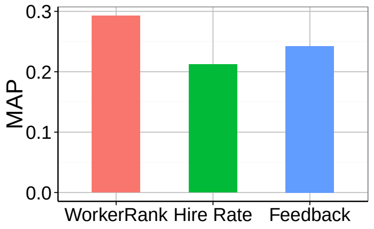

Figure 5: MAP predicting the hiring outcome

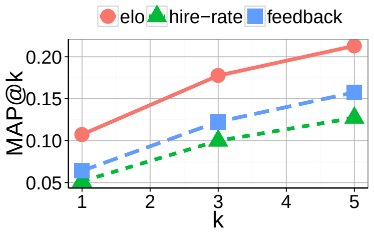

Figure 6: MAP@k predicting the hiring outcome

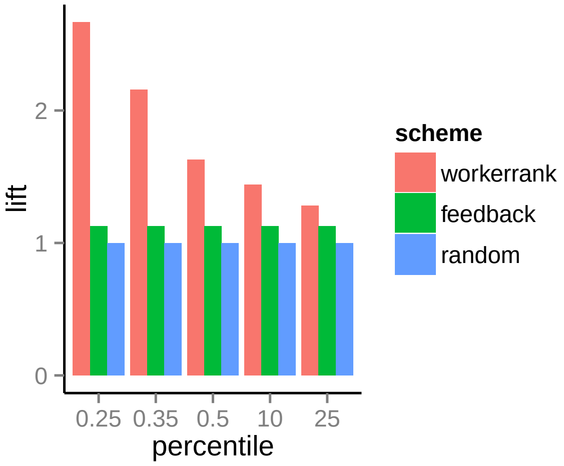

Figure 7: Lift in predicting the hiring outcome

ranking (or, explicit reputation, respectively). Overall, the new algorithm improves the chances of identifying good workers in the top results by 38% (20.8%, respectively).

Lift

To evaluate the quality of the WorkerRank scores, we compare them with the explicit reputation scores as signals for taking hiring decisions. We use the data of the first 52 weeks of our dataset to calculate the WorkerRank scores, and we then use these scores as predictors for the hiring outcomes of the applications submitted during the 53th week. In particular, we rank all of the applications by the WorkerRank scores of the applicants. We then calculate the hiring *lift* in the top x percent of the applications as follows:

$$lift(x) = \frac{\text{hiring probability in the top-}x\%\text{ applicants}}{\text{hiring probability across all applicants}} \qquad (6)$$

The lift shows the performance of WorkerRank versus the performance of a random scoring of the applicants. Similarly, we calculate the lift for the existing feedback-based reputation system and we present the results in the barplot of Figure 7. The plot has five pairs of bars and each pair looks at a different percentage value $x \in \{0.25, 0.35, 0.5, 10, 25\}$. The green bars look at the Elo ratings obtained by WorkerRank and the red bars look at the feedback ratings. The height of each bar shows the lift value for the corresponding scheme and x value. For example, the first green bar from the left shows that the top-0.25 of applicants as ranked by the Elo ratings are 2.66 times more likely to be hired than a random applicant. Note that the lift of the feedback reputation scores is flat at 1.1 for all x values, since the top 25% of the applications correspond to workers with perfect 5-score rating. As a result, the existing reputation system does not provide a sufficient signal to discriminate high quality workers. On the other hand, WorkerRank Elo ratings yield an increasing lift as we limit the percentage of the top-x applications that we consider. The Elo lift is already higher than the feedback lift for $x = 25\%$ and it exceeds 2.5 as we limit x to the top 2.5% of the applications. The results show that the WorkerRank Elo-based reputation system provides a more accurate signal for the contactor application success rather than the existing feedback-based system.

4.4.2 Performance across Job Categories

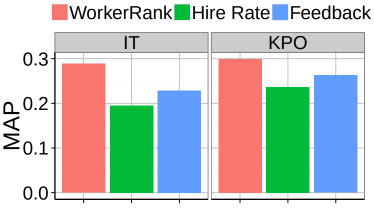

Figure 8: MAP across KPO vs IT jobs

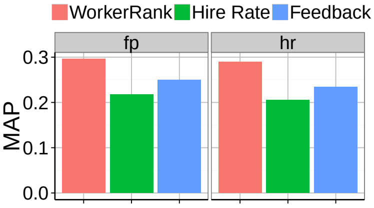

Figure 9: MAP across fixed-price vs hourly jobs

We also illustrate how WorkerRank outperforms the baseline approaches when studied in category subsets of the datasets. Figure 8 shows how the algorithms perfom when applied on the different types of jobs, Knowledge Processing Outsourcing (KPO) and Information Technology (IT). Figure 9 shows how the algorithms perform when applied on fixed-price (fp) versus hourly-rate (hr) jobs.

4.4.3 *Hybrid: combine WorkerRank and Explicit Reputation*

In Figures 10(a) and 10(b) we show how the hybrid approach performs at MAP and map@k: for $k \in [1, 5]$, compared to the two approaches it combines; WorkerRank and explicit reputation. The hybrid model appears to slightly improve WorkerRank, the best of two approaches, although the improvement is not as high as it was in the comparison between implicit versus explicit reputation in Figures 5 and 6.

Figures 11(a), 11(b), and 12 show how the hybrid model behaves in different job categorizations. It is interesting to mention that in all cases the hybrid model improves WorkerRank and explicit feedback, except for the hourly rate jobs and the Web Development category, where algo marginally outperforms the hybrid model and significantly outperforms the explicit reputation models.

Finally, we tested a few weighting combinations to prioritize the influence of one of the two lists during rank aggregation. Equal weights on the two lists appears to be the best combnation that makes the hybrid model behave optimally. The Figures shown for the hybrid model performance assume equal weight on the two lists.

4.4.4 *Skill-wise WorkerRank vs WorkerRank*

In Figure 13 we show how the skill-wise approach performs at MAP compared to the WorkerRank approach. As mentioned earlier, skill-wise WorkerRank produces specialized scores for workers, pertaining specifically to the skills that a potential job would require. The results show that ranking workers by skill-wise scores is more accurate than ranking them based on the WorkerRank scores.

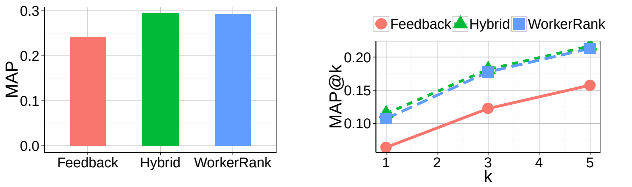

Figure 10: MAP@Inf, MAP@k for hybrid model in predicting the hiring outcome

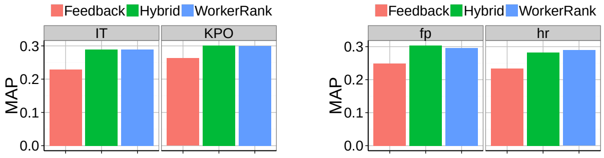

Figure 11: MAP across different job categorizations - hybrid model

In particular, the improvement shown in the Figures of MAP provides a better system to rank workers. However in certain cases skill-wise scores do not appear experimentally to differ from WorkerRank scores. That is because for several jobs a single skill is specified instead of a set of skills. This is one of the reasons why skill scores do not add knowledge to our estimations about workers quality.

We apply logistic regression to combine skill-wise scores in a weighted fashion such that the intersection of our knowledge about the ability of the workers on multiple scores is incorporated. We observe that this improvement was consistent across different job types (hourly and fixed price) and marketplace segments (KPO and IT).

5. RELATED WORK

The problem tackled in this paper overlaps with four research fields; graph link analysis, building online reputation systems, game competition match anaysis and predicting accurate/high-quality response in query-answering.

Graph link analysis research is related to our work, since we represent our data using a bipartite graph and we perform link analysis to examine the job applications of workers along with the respective employer feedback (edge weight). Several approaches have

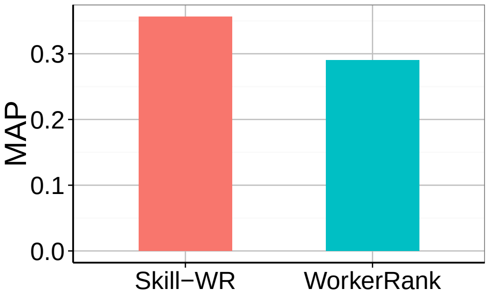

Figure 13: MAP in skill-wise WorkerRank

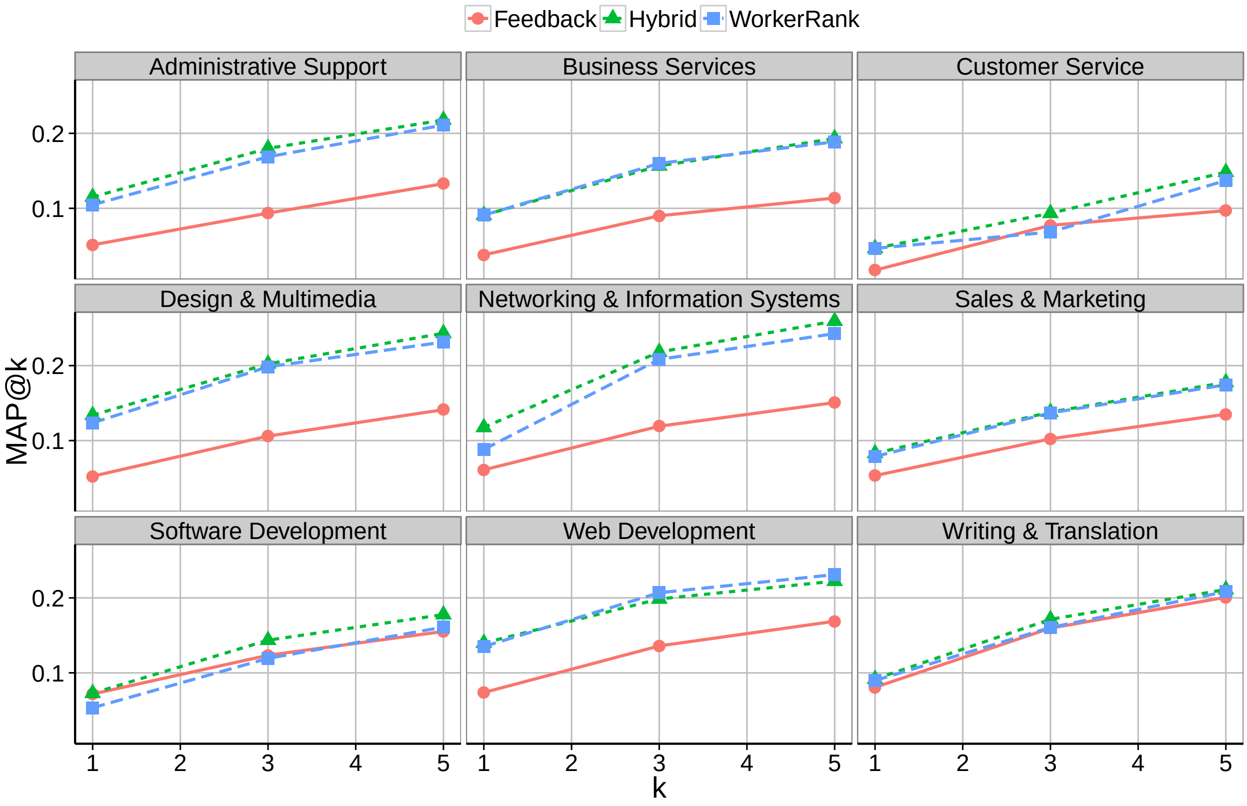

Figure 12: MAP across different job categories - hybrid model

Table 5: Performance Improvement: Hybrid vs WorkerRank vs Explicit Reputation

	Explicit Reputation	WorkerRank	Hybrid	% Improvement	
	RB	WR	HB	HB vs WR	HR vs RB
MAP@1	0.06	0.11	0.12	**+9.1%**	**+100.0%**
MAP@3	0.12	0.18	0.19	**+5.6%**	**+58.33%**
MAP@5	0.15	0.21	0.22	**+4.8%**	**+46.7%**
MAP@Inf	0.24	0.29	0.30	**+3.5%**	**+25.0%**

been proposed about ranking graph nodes in a network, such as PageRank [26], [6] and HITS [19], while Donato et. al. extend the study of HITS in [13] and Zhang et. al. in [33] study how PageRank and HITS perform when applied on the Java forum domain. Finally, Mishra et.al. [24], and Lescovec et. al. in [22] and [21], present their node scoring methods with the presence of both positive and negative edge weights. Note that in our approach we also implicitly make use of negative information about applications, such as the "ignore", "hire" and "reject" feedback responses by the employer. However we only account for the relativity among different feedback labels, that is, who won over whom, hence we do not face restrictions of edge positivity in order to achieve algorithm convergence.

Research on online reputation systems is directly related to our work, as we build a reputation system for workers in the labor marketplace, and we derive additional heuristic de-bias scores for employers. In [11] and [12], Dellarocas summarizes online reputation mechanisms and challenges they face in terms of usage and evaluation. Kokkodis et.al. in [20] discuss how to address data sparseness in building labor marketplace reputation systems. What is more, Archack in [4] discusses how reputation challenges strategic behavior of contestants in the TopCoder marketplace, while Chen in [8] describes their de-biasing mechanism for building a reputation system in a comments rating environment. TwitterRank [32] is another reputation system which aims to build reputation scores such that they incorporate a measure of influence for the Twitter users. Finally, in our past appproaches in [10] and [9], we discuss the usage of link analysis using weighting schemes in order to build reputation systems.

It is interesting to reference a few game competition works such as the Elo method [14] that we are using in our current approach in order to predict the expected hire probability of each worker given our prior knowledge about their opponent's performance and their own performance. Elo is using a Bayesian update scheme to score chess game players based on past matches activity and update their scores by their expected performance in future tournaments. Glickman in [15] presents an improved approach, which keeps updating the mean and variance of the player scores such that confidence information is also carried along with the player's quality estimation. Methods tackling further improvement of match updates are proposed, such as TrueSkill [16] which tackles multi-player and multi-team challenges and Nikolenko et.al. who further improve TrueSkill's challenges of multiway ties and variable team size. Finally, Sismanis in [29] proposes a re-visit on the Elo method which incorporates tournament recency and other parameters in the tournament analysis in order to avoid over-fitting of the player ratings.

Moreover, query answering methods are referenced since they tackle the challenge of predicting quality of a response content such as question answers and social media content; that challenge is similar to our work's goal of predicting worker quality scores. Several approaches have been proposed aiming to identify quality in social media content such as Agichtein et. al. [2] and Bian et. al. [5]. Shah et.al. [28], Suryanto et.al. [30], Jurczyk et. al. [18] and Anderson et. al. [3] study quality of answers in question answering, while Tsaparas et.al. [31], [23] study quality of online review systems, such as in Yelp or Epinions. Finally, Chen et. al. [7] study debiasing approaches to set votes more informative in question-answering systems towards higher quality in answer and expert ranking.

6. CONCLUSIONS

The results of our experiments show that WorkerRank improves recommendation of candidates compared to baseline approaches, since its reputation scores reflect worker quality more accurately. What is more, WorkerRank solves the basic problems encountered in explicit reputation systems (unreliable employer ratings, limited coverage of worker scores, cold start problem for new workers with no history information). Our future work includes research on weighting schemes as discussed in[10] and modeling implicit actions on the marketplace website. In next steps we would extend our study on content-based approaches similar to studying reputation in a skill-wise fashion, such that we turn WorkerRank's collaborative filtering perspective into a higher performance hybrid recommender scheme.

7. REFERENCES

[1] G. Adomavicius and A. Tuzhilin. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE Trans. on Knowl. and Data Eng.*, 17(6):734–749, June 2005.

[2] E. Agichtein, C. Castillo, D. Donato, A. Gionis, and G. Mishne. Finding high-quality content in social media. In *Proceedings of the 2008 International Conference on Web Search and Data Mining*, WSDM '08, pages 183–194, New York, NY, USA, 2008. ACM.

[3] A. Anderson, D. Huttenlocher, J. Kleinberg, and J. Leskovec. Discovering value from community activity on focused question answering sites: A case study of stack overflow. In *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '12, pages 850–858, New York, NY, USA, 2012. ACM.

[4] N. Archak. Money, glory and cheap talk: Analyzing strategic behavior of contestants in simultaneous crowdsourcing contests on topcoder.com. In *Proceedings of the 19th International Conference on World Wide Web*, WWW '10, pages 21–30, New York, NY, USA, 2010. ACM.

[5] J. Bian, Y. Liu, D. Zhou, E. Agichtein, and H. Zha. Learning to recognize reliable users and content in social media with coupled mutual reinforcement. In *Proceedings of the 18th International Conference on World Wide Web*, WWW '09, pages 51–60, New York, NY, USA, 2009. ACM.

[6] S. Brin and L. Page. The anatomy of a large-scale hypertextual web search engine. In *Proceedings of the Seventh International Conference on World Wide Web 7*, WWW7, pages 107–117, Amsterdam, The Netherlands, The Netherlands, 1998. Elsevier Science Publishers B. V.

[7] B.-C. Chen, A. Dasgupta, X. Wang, and J. Yang. Vote calibration in community question-answering systems. In *Proceedings of the 35th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '12, pages 781–790, New York, NY, USA, 2012. ACM.

[8] B.-C. Chen, J. Guo, B. Tseng, and J. Yang. User reputation in a comment rating environment. In *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '11, pages 159–167, New York, NY, USA, 2011. ACM.

[9] M. Daltayanni and L. de Alfaro. Recommending workers in the labor marketplace. *In Workshop: Data Design for Personalization: Current Challenges and Emerging Opportunities. In conjunction with WSDM 2014.*

[10] M. Daltayanni, L. de Alfaro, P. Papadimitriou, and P. Tsaparas. Reputation system for online labor marketplaces. *In Proceedings of EDBT/ICDT 2014.*

[11] C. Dellarocas. The digitization of word of mouth: Promise

and challenges of online feedback mechanisms. *Manage. Sci.*, 49(10):1407–1424, Oct. 2003.

[12] C. Dellarocas. Reputation mechanisms. In *Handbook on Economics and Information Systems*, page 2006. Elsevier Publishing, 2006.

[13] D. Donato, S. Leonardi, and P. Tsaparas. Stability and similarity of link analysis ranking algorithms. *Internet Mathematics*, 3(4):479–507, 2007.

[14] A. E. Elo. *The rating of chessplayers, past and present*. Arco Pub., New York, 1978.

[15] M. E. Glickman. The glicko system. 1995.

[16] R. Herbrich, T. Minka, and T. Graepel. Trueskilltm: A bayesian skill rating system. In B. Schoelkopf, J. Platt, and T. Hoffman, editors, *NIPS*, pages 569–576. MIT Press, 2006.

[17] N. Hu, J. Zhang, and P. A. Pavlou. Overcoming the j-shaped distribution of product reviews. *Commun. ACM*, 52(10):144–147, Oct. 2009.

[18] P. Jurczyk and E. Agichtein. Discovering authorities in question answer communities by using link analysis. In *Proceedings of the Sixteenth ACM Conference on Conference on Information and Knowledge Management*, CIKM '07, pages 919–922, New York, NY, USA, 2007. ACM.

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

[20] M. Kokkodis and P. G. Ipeirotis. Have you done anything like that?: predicting performance using inter-category reputation. In S. Leonardi, A. Panconesi, P. Ferragina, and A. Gionis, editors, *WSDM*, pages 435–444. ACM, 2013.

[21] J. Leskovec, D. Huttenlocher, and J. Kleinberg. Predicting positive and negative links in online social networks. In *Proceedings of the 19th international conference on World wide web*, WWW '10, pages 641–650, New York, NY, USA, 2010. ACM.

[22] J. Leskovec, D. Huttenlocher, and J. Kleinberg. Signed networks in social media. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '10, pages 1361–1370, New York, NY, USA, 2010. ACM.

[23] Y. Lu, P. Tsaparas, A. Ntoulas, and L. Polanyi. Exploiting social context for review quality prediction. In *Proceedings of the 19th International Conference on World Wide Web*, WWW '10, pages 691–700, New York, NY, USA, 2010. ACM.

[24] A. Mishra and A. Bhattacharya. Finding the bias and prestige of nodes in networks based on trust scores. In *Proceedings of the 20th international conference on World wide web*, WWW '11, pages 567–576, New York, NY, USA, 2011. ACM.

[25] oDesk. https://www.odesk.com.

[26] L. Page, S. Brin, R. Motwani, and T. Winograd. The pagerank citation ranking: Bringing order to the web, 1999.

[27] V. Pihur, S. Datta, and S. Datta. Weighted rank aggregation of cluster validation measures. *Bioinformatics*, 23(13):1607–1615, July 2007.

[28] C. Shah and J. Pomerantz. Evaluating and predicting answer quality in community qa. In *Proceedings of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '10, pages 411–418, New York, NY, USA, 2010. ACM.

[29] Y. Sismanis. How i won the "chess ratings - elo vs the rest of the world" competition. *CoRR*, abs/1012.4571, 2010.

[30] M. A. Suryanto, E. P. Lim, A. Sun, and R. H. L. Chiang. Quality-aware collaborative question answering: Methods and evaluation. In *Proceedings of the Second ACM International Conference on Web Search and Data Mining*, WSDM '09, pages 142–151, New York, NY, USA, 2009. ACM.

[31] P. Tsaparas, A. Ntoulas, and E. Terzi. Selecting a comprehensive set of reviews. In *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '11, pages 168–176, New York, NY, USA, 2011. ACM.

[32] J. Weng, E.-P. Lim, J. Jiang, and Q. He. Twitterrank: Finding topic-sensitive influential twitterers. In *Proceedings of the Third ACM International Conference on Web Search and Data Mining*, WSDM '10, pages 261–270, New York, NY, USA, 2010. ACM.

[33] J. Zhang, M. S. Ackerman, and L. Adamic. Expertise networks in online communities: Structure and algorithms. In *Proceedings of the 16th International Conference on World Wide Web*, WWW '07, pages 221–230, New York, NY, USA, 2007. ACM.