Contests for Predictive Algorithms: Ensembling, Interdependency, and Optimal Rewards Design

Mehmet Eren Ahsen
Department of Business Administration
Department of Biomedical and Translational Sciences
University of Illinois Urbana-Champaign
ahsen@illinois.edu

Srinivasan Raghunathan
Naveen Jindal School of Management
University of Texas at Dallas
sraghu@utdallas.edu

Mehmet U.S. Ayvaci
Naveen Jindal School of Management
University of Texas at Dallas
mehmet.ayvaci@utdallas.edu

Ramanath Subramanyam
Department of Business Administration
University of Illinois Urbana-Champaign
rsubrama@illinois.edu

Abstract

With the increased availability of data and widespread use of AI/ML technologies, firms are increasingly using crowdsourcing contests to acquire predictive algorithms for business use. A unique feature of these contests is that the designer can create an ensemble of submitted algorithms and improve upon the predictive accuracy (or quality) of individual algorithms. Given this departure from conventional contests, we ask how the optimal rewarding schemes should be in the presence of ensembling and interdependent algorithms. To answer this question, we develop a stylistic model of a contest for a predictive algorithm with two participants. As opposed to the single best contribution typically sought out in conventional contests and the winner-takes-all reward scheme, we find that both the winner and the runner-up could receive rewards in contests for algorithms, depending on whether the ensembled algorithms substitute or complement each other as determined by the algorithm interdependency. We show that the degree of substitution or complementarity can fundamentally alter the structure of the optimal reward scheme. We highlight our results using a real-world algorithm contest for predicting breast cancer using digital mammograms, we demonstrate the practical applicability of our framework.

Keywords: Crowdsourcing Contests, Algorithms, Ensemble, Competition Design, Artifical Intelligence

1. Introduction

Advances in artificial intelligence/machine learning (AI/ML) and vast increases in the scale and scope of available data have boosted firms' reliance on algorithms to operate efficiently and compete effectively (Iansiti and Lakhani, 2020). In particular, predictive algorithms have become ubiquitous within firms across industries. Firms can use predictive algorithms for a variety of purposes that include making product recommendations (Adomavicius et al., 2018), refining search results (Song et al., 2019), and consumer targeting (Kadiyala et al., 2021). Similarly, financial services firms employ algorithms to predict customer risk and design customized products, among others. In the healthcare sector, automated health risk prediction systems enable hospitals to deliver timely and life-saving care to patients (Adjerid et al., 2023), achieve operational efficiency (Ahsen et al., 2023), and enhance clinical outcomes (Ahsen, Ayvaci, and Raghunathan, 2019).

Despite the potential value of predictive algorithms, developing them in-house may be challenging for some firms for various reasons. For instance, firms may not have internal expertise in predictive modeling techniques (e.g., deep learning). Training existing employees or hiring new ones with the required expertise could be costly. Developing predictive algorithms may also require specialized infrastructure (e.g., technologies to operate big data) and costly investments in hardware and software. Moreover, the preponderance of predictive modeling techniques implies that the expertise in developing prediction algorithms is likely to be dispersed among a large group of individuals. Consequently, firms increasingly utilize crowdsourcing contests in which they seek algorithms from outside contestants and wisdom of crowds to solve their prediction problems (Chesbrough et al., 2006; Jeppesen and Lakhani, 2010). Essentially, crowdsourcing contests can help firms extend their internal capability, account for internal deficiencies in data analytics, and acquire algorithmic innovations.

There is extensive literature on reward design for contests and tournaments that involve tasks such as prototype and design (e.g., logo design contest), ideation (e.g., new product ideas), and art (e.g., photography contests) (Cohen et al., 2008; Moldovanu and Sela, 2001; Terwiesch and Xu, 2008). However, contests for algorithms differ from those examined in the extant literature in one key dimension. In algorithm contests, the designer can combine two or more of the submitted algorithms to produce an ensemble that is superior to any of the individual algorithms. The ensembling procedure could range from a simple aggregation (e.g.,

URI: https://hdl.handle.net/10125/107127
978-0-9981331-7-1

weighted average) of outputs of individual algorithms to one that merges the logic of algorithms to create a new one. In contrast, combining multiple submissions to create a superior solution may not be viable or feasible in contexts studied by traditional contest literature. If two prediction algorithms use very similar approaches (e.g., linear regression), their predictions will likely be highly positively correlated, and the incremental value from combining them is likely to be limited even if each algorithm has a high predictive power, when considered individually. On the other hand, the incremental value from combining two algorithms that use vastly different approaches (e.g., linear regression and deep learning) could be very high despite low individual predictive power. Moreover, contests for algorithms seem to induce contestant participation even when they do not offer any extrinsic rewards, suggesting that contestants may enjoy intrinsic utility from tackling the problems posed by such contests. These unique features of algorithms make the design of crowdsourcing contests for algorithms particularly novel and challenging. For example, answers to specific questions such as (i) should a contest designer offer a reward only to the winner as is done in many traditional contests or should there be a reward offered to the winner as well as the runner-up, (ii) what should be the relative reward amounts if the designer rewards multiple contestants, and (iii) how does the similarity between the contestants' solutions affect the optimal rewards and contest outcomes such as the predictive power of the ensemble and designer's overall payoff, are unclear.

To obtain insights into the optimal reward design and outcomes of crowdsourcing contests for algorithms and highlight the differences between them and other contests, we first model a traditional contest with two participants. In the traditional contest, the contest designer cannot create an ensemble of the submissions. Hence, the designer derives value only from one winning submission. Then, we model a contest for algorithms for a prediction problem in which the designer can combine multiple submitted algorithms into an *ensemble*. We compare the optimal reward schemes in the two cases to highlight the differences between the reward schemes in traditional contests and contests involving ensembled algorithms. We then examine how key factors such as the correlation between the predictions of ensembled algorithms affect the contest design and outcomes. Finally, we supplement our theoretical insights with results from a computational study that uses real-life data, which come from an algorithmic contest for mammogram-based breast cancer prediction.

2. Model Setup and Theoretical Analysis

We examine reward schemes for crowdsourced algorithm contests for prediction problems. In this context, a designer ("she") needs an algorithm to solve a prediction problem she faces. She sets up a reward scheme to incentivize agents ("he") to exert efforts to develop an algorithm. For analytical tractability and expositional clarity, we consider a contest with two agents. Essentially, the purpose of rewards in our model is to induce agent efforts as opposed to agent participation in the contest (see Körpeoğlu and Cho, 2018 for the interrelationhip between rewards and agents' effort trumping the effects of rewards on participation). We denote the two agents as A_1 and A_2 and the quality of the algorithm submitted by agent A_i as s_i. The quality could be defined as the algorithm's performance.

The designer announces a reward of k_W per unit quality to the agent ("winner") who submits the higher quality algorithm and k_L per unit quality to the agent ("runner-up") who submits the lower quality algorithm. We assume $k_W \geq k_L \geq 0$. Thus, if A_1 is the winner, then the designer will offer $k_W s_1$ to him and $k_L s_2$ to the runner-up, A_2. Offering a quality-contingent reward is consistent with the finding that performance-contingent rewards are more effective in inducing desirable efforts than fixed rewards in innovation contests (Terwiesch and Xu, 2008). A_i incurs a cost of $c_i s_i^2/2$ to develop an algorithm of quality s_i. Without loss of generality, we assume $c_1 < c_2$. Accordingly, we denote A_1 as the low-cost agent and A_2 as the high-cost agent. Moreover, we focus on a pure strategy Nash equilibrium which exists only when $c_1 < c_2/2$. This condition implies that the agents' costs are sufficiently differentiated so that it is unprofitable for a high-cost agent to mimic the low-cost agent in equilibrium.

It is well known that the agents in innovation contests are motivated not only by the reward amount but also by their intrinsic motivation (Kaufmann et al., 2011). The intrinsic motivation could be driven by factors such as peer recognition, academic publications, potential for learning and skill enhancement, and future job prospects. We model the intrinsic motivation as the additional utility an agent derives from developing an algorithm for the contest. Specifically, A_i accrues an *intrinsic utility* of ωs_i. The value of ω will likely depend on the problem being solved by the algorithm. For example, a contest that addresses a more interesting, high impact, and challenging problem could have a higher ω for agents.

The designer's payoff from using an algorithm is increasing in its quality. Specifically, we assume that the designer's payoff is linearly increasing in quality, with the coefficient normalized to one without loss of generality. Thus, the designer's payoff from using only algorithm i is s_i. If the designer creates an ensemble of the two algorithms, we assume that the ensemble quality and hence the designer's payoff is given by

$$s_e = s_1 + s_2 - \lambda s_1 s_2, \tag{1}$$

The parameter λ in Equation (1) performs the same role as correlation coefficient between two algorithms in the sense when λ is positive (resp., negative), the two algorithms are complements (resp., substitutes). An increase in the absolute value $|\lambda|$ increases the *degree of substitution* between the two algorithms if λ is positive and the *degree of complementarity* between the two algorithms if λ is negative. We further assume that $\lambda < \min\{\frac{1}{s_1}, \frac{1}{s_2}\}$ in equilibrium. This assumption ensures that the ensemble quality is never inferior to that of either one of the algorithms. Moreover, λ is assumed to be not too negative to ensure an interior solution for the equilibrium reward. Technically, these restrictions point to the assumption $-(c_1+c_2) < \lambda < \frac{2c_1}{1+\omega}$. We also assume $\omega < 1$ so that the designer's payoff from an algorithm is higher than the agent's intrinsic utility from developing the algorithm and hence the designer offers a positive reward at least to the winner. Finally, we assume that the designer incurs a negligible cost to create the ensemble in the main model.

The sequence of events for the contest is as follows. In the first stage, the designer determines the reward scheme. In the second stage, agents exert efforts, develop their algorithms, and they submit them. In the third stage, the designer assesses the algorithms, creates the ensemble, determines the winner, and offers the rewards as per the reward scheme determined in stage 1. We solve for a pure strategy sub game perfect Nash equilibrium using backward induction. We assume that the designer knows the agents' intrinsic utility and costs, and the correlation parameter λ prior to announcing the rewards.

We use two benchmark cases to highlight the differences in the optimal reward schemes between contests for algorithms and conventional contests. In the *first benchmark*, we consider contests in which the designer is forced to use only the higher quality submission, possibly because combining submissions from two different agents is too costly. For example, in a logo design contest, creating an ensemble design using the desirable features of individual designs could require substantial additional and possibly duplicate effort. Even in the case of contests for algorithms, merging the logic of individual algorithms to create an ensemble could require significant effort (e.g., see Ahsen, Vogel, et al., 2019). This benchmark represents the cases in which ensemble is "infeasible." In the *second benchmark*, we consider contests in which the designer creates an ensemble of the two submissions (i.e., ensemble is "feasible"). However, in this benchmark, the designer uses the"winner-takes-all" reward scheme, which is a conventional reward scheme used widely in practice and examined extensively in the academic literature (Halac et al., 2017). After analyzing these two benchmarks, we study the *main model* corresponding to the optimal reward design. In the main model, the designer creates an ensemble similar to the second benchmark, but employs a reward scheme which does not limit the reward only to the winner.

2.1. Contests Where Ensemble is Infeasible

In this benchmark, *the designer does not create an ensemble*. Let the designer's payoff be R_D and that of agent A_i be R_i, $i \in \{1,2\}$. Then, we have the following if A_1 is the winner.

$$\begin{aligned} R_D &= s_1 - k_W s_1 - k_L s_2, \\ R_1 &= (k_W+\omega)s_1 - 0.5c_1 s_1^2, \\ R_2 &= (k_L+\omega)s_2 - 0.5c_2 s_2^2. \end{aligned} \tag{2}$$

If A_2 is the winner, then we can obtain the designer's and the agents' payoff expressions by interchanging k_W and k_L in Equation (2). We use superscript $b1$ in the equilibrium solutions for the case when ensemble is infeasible.

Lemma 2.1 *The following rewards and qualities constitute a unique pure strategy Nash equilibrium when ensemble is infeasible:*

$$k_W^{b1} = \frac{1-\omega}{2},\ k_L^{b1} = 0,$$

$$s_1^{b1} = \frac{1+\omega}{2c_1},\ s_2^{b1} = \frac{\omega}{c_2}.$$

Lemma 2.1 reveals that when the designer uses only the higher quality algorithm, the winner-takes-all reward scheme is optimal. The low-cost agent would be the winner, and the contest outcomes depend only on the low-cost agent's parameters. These observations are intuitive in light of the fact that the rewards' purpose is to induce agent effort rather than increase agent participation because intrinsic utility induces all agents to participate regardless of the reward. Using Lemma 2.1, we compute the designer's and agents' equilibrium payoffs and the final algorithm quality in the first benchmark as follows.

$$R_1^{b1} = \frac{(1+\omega)^2}{8c_1},\ R_2^{b1} = \frac{\omega^2}{2c_2},\ R_D^{b1} = \frac{(1+\omega)^2}{4c_1},\ s_e^{b1} = \frac{1+\omega}{2c_1}. \tag{3}$$

As expected, each agent's payoff is decreasing in his own cost and increasing in the intrinsic utility. Also, the designer's payoff as well as the final algorithm quality is decreasing in the low-cost agent's cost and increasing in the agents' intrinsic utility.

2.2. Winner-Takes-All When Ensemble is Feasible

In this benchmark, the designer would create an ensemble of the two algorithms but *only the winner receives a reward*. We have the following payoffs if A_1 is the winner.

$$\begin{aligned} R_D &= s_e - k_W s_1, \\ R_1 &= (k_W+\omega)s_1 - 0.5c_1 s_1^2, \\ R_2 &= (0+\omega)s_2 - 0.5c_2 s_2^2. \end{aligned} \tag{4}$$

We can obtain the designer's and agents' payoffs if A_2 is the winner in an analogous manner. We use superscript $b2$ for the equilibrium solutions in the second benchmark.

Lemma 2.2 *The following rewards and qualities constitute a unique pure strategy Nash equilibrium under the winner-takes-all reward scheme when ensemble is feasible:*

(i) If $\lambda \le \frac{c_2(1-\omega)}{\omega}$,

$$k_W^{b2} = \frac{1-\omega}{2} - \omega\frac{\lambda}{2c_2},\ k_L^{b2} = 0,$$

$$s_1^{b2} = \frac{(c_2\omega + c_2 - \lambda\omega)}{2c_1 c_2},\ s_2^{b2} = \frac{\omega}{c_2}.$$

(ii) otherwise,

$$k_W^{b2}=0,\, k_L^{b2}=0,$$

$$s_1^{b2}=\frac{\omega}{c_1},\, s_2^{b2}=\frac{\omega}{c_2}.$$

A comparison of the solutions in the two benchmarks reveals the following. While only the winner gets a reward in both benchmarks, the designer respectively offers a *smaller* or a *larger* reward to the winner (in the amount of $-\omega\frac{\lambda}{2c_2}$) when she creates an ensemble than when she does not, depending on whether the two algorithms are *substitutes* or *complements*. This result arises because, when the algorithms are ensembled, the low-quality (i.e., high-cost agent's) algorithm diminishes (enhances) the marginal value of the high-quality (i.e., low-cost agent's) algorithm to the designer when they are substitutes (complements). On the other hand, there is no such interdependency between the algorithms if they are not ensembled. Hence, the designer offers a smaller reward to the winner (inducing less effort) in the case of substitutes and a larger reward to the winner (inducing more effort) in the case of complements if the algorithms are ensembled, as compared to the case when the ensemble is infeasible. We also observe that the reduction (increase) in reward is increasing in the degree of substitution (complementarity) and in ω, and decreasing in c_2. This is because an increase in either the degree of substitution (complementarity) or ω amplifies (mitigates) the reduction (increase) in the marginal value of the high-quality algorithm, but an increase in the high-cost agent's cost has the opposite effect on the marginal value of the high-quality algorithm to the designer. Finally, when the degree of substitution is greater than a threshold and the agents have a high intrinsic utility, the designer does not offer any reward when she can create the ensemble (i.e., second benchmark case) even though she pays the winner when she cannot create the ensemble (i.e., first benchmark case). Essentially, when the ensemble is feasible, the relatively high quality produced even by the high-cost agent on his own (without any reward) when the intrinsic utility is high makes the marginal utility of reward negative for the designer if the degree of substitution is high. Consequently, she does not find it profitable to offer a positive reward even to the winner under such conditions.[1] Using Lemma 2.2, we compute the designer's and the agents' equilibrium payoffs and the equilibrium ensemble quality for the second benchmark case as follows.

(i) If $\lambda\leq\frac{c_2(1-\omega)}{\omega}$

$$\begin{cases} R_1^{b2}=\frac{(c_2\omega+c_2-\lambda\omega)^2}{8c_1c_2^2}, R_2^{b2}=\frac{\omega^2}{2c_2},\\ R_D^{b2}=\frac{2c_2\omega(2c_1+c_2-\lambda)+c_2^2+\omega^2(c_2-\lambda)^2}{4c_1c_2^2},\\ s_e^{b2}=\frac{c_2\omega(2c_1+c_2-2\lambda)+c_2^2+\lambda\omega^2(\lambda-c_2)}{2c_1c_2^2}, \end{cases}$$

(ii) otherwise,

$$\begin{cases} R_1^{b2}=\frac{\omega^2}{2c_1}, R_2^{b2}=\frac{\omega^2}{2c_2}, R_D^{b2}=\frac{\omega(c_1+c_2-\lambda\omega)}{c_1c_2},\\ s_e^{b2}=\frac{\omega(c_1+c_2-\lambda\omega)}{c_1c_2}. \end{cases}$$

A comparison of the outcomes in the two benchmark cases reveals that the designer, under the winner-takes-all reward scheme, has a higher pay-off when the ensemble is feasible than when the ensemble is infeasible. While the high-cost agent realizes the same payoff (because of zero rewards in both benchmarks), the low-cost agent realizes a lower payoff under the winner-takes-all reward scheme when the designer can create the ensemble than under the case when the ensemble is infeasible. Essentially, the designer takes advantage of her ability to create an ensemble by improving the overall algorithm quality while paying less to the agents.

2.3. Optimal Rewards When Ensemble is Feasible

We identify the optimal reward scheme for algorithm contests by allowing the designer to offer rewards to both agents, i.e., k_W to the winner and k_L to the runner-up. In such contests, the designer can create an ensemble with quality s_e. The payoffs when A_1 is the winner are as follows.

$$R_D=s_e-k_Ws_1-k_Ls_2,$$

$$R_1=(k_W+\omega)s_1-0.5c_1s_1^2,$$

$$R_2=(k_L+\omega)s_2-0.5c_2s_2^2. \tag{8}$$

We can obtain the designer's and agents' payoffs if A_2 is the winner in an analogous manner. We use superscript * to indicate the equilibrium solutions under the optimal reward scheme.

Lemma 2.3 *The following rewards and qualities constitute a unique pure strategy Nash equilibrium in algorithmic contests: A. If the algorithms are substitutes (i.e., $\lambda\geq 0$), then*

(i) If $0\leq\lambda\leq\frac{c_2(1+\omega)-\sqrt{c_2^2(1+\omega)^2-8c_1c_2(1-\omega)\omega}}{2\omega}$,

$$k_W^*=\frac{2c_1c_2-c_1\lambda}{4c_1c_2-\lambda^2}+\omega\frac{\lambda^2-c_1\lambda-2c_1c_2}{4c_1c_2-\lambda^2},$$

$$k_L^*=\frac{2c_1c_2-c_2\lambda}{4c_1c_2-\lambda^2}+\omega\frac{\lambda^2-c_2\lambda-2c_1c_2}{4c_1c_2-\lambda^2}. \tag{9}$$

$$s_1^*=\frac{(1+\omega)(2c_2-\lambda)}{4c_1c_2-\lambda^2},\quad s_2^*=\frac{(1+\omega)(2c_1-\lambda)}{4c_1c_2-\lambda^2}.$$

[1] Note that if $\omega>1$ and sufficiently high then the designer could offer a zero reward even when the ensemble is not feasible, i.e., in the first benchmark. However, the ability to create an ensemble as in the second benchmark could induce the designer to offer a zero reward even though she would not do so when the ensemble is not feasible as in the first benchmark.

(ii) If $\frac{c_2(1+\omega)-\sqrt{c_2^2(1+\omega)^2-8c_1c_2(1-\omega)\omega}}{2\omega}<\lambda\leq\frac{c_2(1-\omega)}{\omega}$,

$$k_W^*=\frac{1-\omega}{2}-\omega\frac{\lambda}{2c_2},\ k_L^*=0.$$

$$s_1^*=\frac{1+\omega}{2c_1}-\omega\frac{\lambda}{2c_1c_2},\quad s_2=\frac{\omega}{c_2}.$$

(iii) Otherwise,

$$k_W^*=0,\ k_L=0,$$

$$s_1^*=\frac{\omega}{c_1},\quad s_2=\frac{\omega}{c_2}.$$

B. If the algorithms are complements (i.e., $\lambda<0$*), then*

$$k_W^*=k_L^*=\frac{c_1+c_2}{2(c_1+c_2+\lambda)}-\omega\frac{c_1+c_2+2\lambda}{2(c_1+c_2+\lambda)},$$

$$s_1^*=\frac{(1+\omega)(c_1+c_2)}{2c_1(c_1+c_2+\lambda)},\ s_2^*=\frac{(1+\omega)(c_1+c_2)}{2c_2(c_1+c_2+\lambda)}.$$

We make the following observations from Lemma 2.3

(a) The optimal reward scheme is structurally different when the algorithms are substitutes and when they are complements. Notably, the optimal reward per unit quality is the same for both agents when the algorithms are complements. On the other hand, when the algorithms are substitutes, it is optimal to offer differential reward per unit quality to the winner and the runner-up unless the degree of substitution is high in which case the designer should reward neither agent.

(b) When the algorithms are complements or when the degree of substitution between algorithms is below a threshold (i.e., $\lambda\leq\frac{c_2(1+\omega)-\sqrt{c_2^2(1+\omega)^2-8c_1c_2(1-\omega)\omega}}{2\omega}$), the designer rewards both agents, which is in contrast to the traditional winner-takes-all reward scheme shown to be optimal if ensemble is infeasible. On the other hand, if the degree of substitution is above the threshold (but not too high), then the winner-takes-all reward scheme is optimal even when the ensemble is feasible.

(c) The designer rewards the winner (weakly) more per unit quality than she does the runner-up ($k_W\geq k_L$).

(d) As in the winner-takes-all scheme, the low-cost agent is the winner. Yet, unlike the winner-takes-all reward scheme, an agent's algorithm quality depends not only on his own cost but also the cost of the other agent because of the rewards depend on both agents' costs.

(e) When the agents' intrinsic utility and the degree of substitution are high (i.e., $\lambda>\frac{c_2(1-\omega)}{\omega}$), the designer has no incentive to offer a reward to any agent, as in the second benchmark.

(f) An increase in the agents' intrinsic utility reduces the reward regardless of whether the algorithms are complements or substitutes. In the case of substitutes, an increase in contestants' intrinsic utility can switch the reward scheme to one where only the winner or neither receiving the reward. The switch occurs at a lower threshold of intrinsic utility when the degree of substitution is higher.

The above observations imply that if the contributed algorithms are expected to be substitutes, it is optimal for the designer to choose a winner by offering a higher reward per unit quality to the contributor that produces a higher quality algorithm and a lower reward per unit quality to the one that produces a lower quality algorithm, essentially creating a *competitive contest*; on the other hand, if the contributed algorithms are expected to be complements, it is optimal for the designer to not choose a winner and offer the same reward per unit quality regardless of the ordering of quality, creating a *collaborative contest*. Moreover, if the degree of substitution is above the threshold (but not too high), then the competitive contest degenerates to the *winner-takes-all contest* when the ensemble is feasible. The agents' intrinsic utility plays a key role in the determining the type of contest that is optimal. Using Lemma 2.3, we compute the designer's and agents' equilibrium payoffs and the equilibrium quality under the optimal reward scheme as follows.

A. If the algorithms are substitutes,

(i) If $0<\lambda\leq\frac{c_2\omega+c_2-\sqrt{c_2}\sqrt{8c_1(\omega-1)\omega+c_2(1+\omega)^2}}{2\omega}$,

$$\begin{cases} R_1^{b2}=\frac{c_1(2c_2-\lambda)^2(1+\omega)^2}{2(4c_1c_2-\lambda^2)^2},\ R_2^{b2}=\frac{c_2(2c_1-\lambda)^2(1+\omega)^2}{2(4c_1c_2-\lambda^2)^2},\\ R_D^{b2}=\frac{(1+\omega)^2(c_1+c_2-\lambda)}{4c_1c_2-\lambda^2},\\ s_e^{b2}=\frac{(1+\omega)\left(\lambda^3-12c_1c_2\lambda+8c_1c_2(c_1+c_2)-\lambda\omega(2c_1-\lambda)(2c_2-\lambda)\right)}{(4c_1c_2-\lambda^2)^2}. \end{cases}$$

(ii) If $\frac{c_2\omega+c_2-\sqrt{c_2}\sqrt{8c_1(\omega-1)\omega+c_2(1+\omega)^2}}{2\omega}<\lambda\leq\frac{c_2(1-\omega)}{\omega}$

$$\begin{cases} R_1^{b2}=\frac{(c_2\omega+c_2-\lambda\omega)^2}{8c_1c_2^2},\ R_2^{b2}=\frac{\omega^2}{2c_2},\\ R_D^{b2}=\frac{2c_2\omega(2c_1+c_2-\lambda)+c_2^2+\omega^2(c_2-\lambda)^2}{4c_1c_2^2},\\ s_e^{b2}=\frac{c_2\omega(2c_1+c_2-2\lambda)+c_2^2+\lambda\omega^2(\lambda-c_2)}{2c_1c_2^2}. \end{cases}$$

(iii) Otherwise,

$$\begin{cases} R_1^{b2}=\frac{\omega^2}{2c_1},\ R_2^{b2}=\frac{\omega^2}{2c_2},\ R_D^{b2}=\frac{\omega(c_1+c_2-\lambda\omega)}{c_1c_2},\\ s_e^{b2}=\frac{\omega(c_1+c_2-\lambda\omega)}{c_1c_2}. \end{cases}$$

B. If the algorithms are complements,

(iv)

$$\begin{cases} R_1=(1+\omega)^2\frac{(c_1+c_2)^2}{8c_1(c_1+c_2+\lambda)^2},\ R_2=(1+\omega)^2\frac{(c_1+c_2)^2}{8c_2(c_1+c_2+\lambda)^2},\\ R_D=(1+\omega)^2\frac{(c_1+c_2)^2}{4c_1c_2(c_1+c_2+\lambda)}\\ s_e=(1+\omega)(c_1+c_2)^2\left[\frac{2(c_1+c_2)+\lambda(1-\omega)}{4c_1c_2(c_1+c_2+\lambda)^2}\right] \end{cases}$$

2.3.1. Impact of Algorithmic Interdependency The equilibrium outcomes under the optimal reward scheme highlight the key role played by the interdependency between submitted algorithms (or algorithmic interdependency) in determining the equilibrium reward structure, ensemble quality, and designer's and agents' payoffs. Moreover, the agents' intrinsic utility moderates the role of the algorithmic interdependency in the outcomes. We examine these roles next.

In analyzing the role of algorithmic interdependency, we focus on the regions where the designer rewards both agents ,because the equilibria in other regions are identical to those in the second benchmark in which the designer uses the winner-takes-all reward scheme. The following proposition shows the effect of algorithmic interdependency on equilibrium rewards.

Proposition 2.1

(i) If the algorithms are complements, the reward per unit quality and the payoffs are increasing in the degree of complementarity for both the winner and the runner-up.

(ii) If the algorithms are substitutes, the reward per unit quality and the payoff for the runner-up are decreasing in the degree of substitution, and the reward per unit quality and the payoff for the winner are decreasing in the degree of substitution if and only if either $\omega \geq 1/3$ or $\lambda \leq 2(c_2 - \sqrt{c_2(c_2 - c_1)})$,

(iii) The difference between the rewards for unit quality to the winner and the runner-up is increasing in the degree of substitution if the algorithms are substitutes, and independent of the degree of complementarity if they are complements. The difference between total rewards to the winner and the runner-up is increasing in the degree of substitution/complementarity if the algorithms are substitutes/complements.

Proposition 2.1(i) reveals that the designer offers a higher reward—both per unit quality and total—to both agents when the algorithms are more complementary to each other. When the algorithms are complements, the quality of the ensemble algorithm exceeds the sum of qualities of individual algorithms, and an improvement in either algorithm quality enhances the marginal value of the other algorithm's quality to the designer. An increase in the degree of complementarity strengthens this synergistic relationship between the two algorithms. Consequently, the designer is incentivized to offer more rewards to both agents if the degree of complementarity increases. On the other hand, the reverse generally happens when the algorithms become more substitutable (Proposition 2.1(ii)). That is, the designer offers smaller rewards—both per-unit quality and total—to both agents when the algorithms are substitutes except when $\lambda > 2(c_2 - \sqrt{c_2(c_2 - c_1)})$ and $\omega < 1/3$. The opposite effects of degree of complementarity and degree of substitution on rewards arise because the effects of an improvement in one algorithm's quality on the marginal value of the other algorithm's quality to the designer are opposite to each other when the algorithms are complements and when they are substitutes; we observe enhancing effect in the case of complements and diminishing effect in the case of substitutes.

Interestingly, Proposition 2.1(ii) shows that an increase in the degree of substitution can induce the designer to increase the reward to the winner when the agents' intrinsic utility is low (i.e., when $\omega < 1/3$) and the degree of substitution is high (i.e., when $\lambda > 2(c_2 - \sqrt{c_2(c_2 - c_1)})$). This finding implies that an increase in the degree of substitution increases the marginal contribution of the high-quality algorithm but diminishes the marginal contribution of the low-quality algorithm to the designer when these conditions hold. The intuition is as follows. We can verify that an increase in the degree of substitution diminishes the marginal contribution of the low-quality algorithm (or high-cost agent) more than that of the the high-quality algorithm (or low-cost agent), i.e., $|\frac{d^2 s_e}{d\lambda ds_2}| = s_1 > |\frac{d^2 s_e}{d\lambda ds_1}| = s_2$. Consequently, the designer reduces the high-cost agent's (i.e., runner-up's) reward more than the low-cost agent's (i.e., winner's) when the degree of substitution increases. The differential reduction in rewards in turn increases the difference between the qualities of the two algorithms. On the other hand, any decrease in the quality of the runner-up's algorithm mitigates the substitution effect and in turn, increases the marginal contribution of the high-quality algorithm in the ensemble quality. Such an increase in the marginal contribution of the high-quality algorithm induces the designer to incentivize the winner (or the low-cost agent) to exert more effort. When $\lambda > 2(c_2 - \sqrt{c_2(c_2 - c_1)})$ and $\omega < 1/3$, the latter effect dominates the former. Consequently, the designer increases her reliance on the high-quality algorithm to improve the ensemble quality and provides more rewards to the winner to mitigate the adverse effects of algorithm substitutability on ensemble quality under the conditions stated in the proposition.

Finally, the gap between the rewards for the winner and the runner-up is increasing both in the degree of substitution and the degree of complementarity (Proposition 2.1(iii)). When the two algorithms are substitutes, the designer's relative preference for the higher quality algorithm from the low-cost agent over the lower quality algorithm from the high-cost agent is enhanced with an increase in the degree of substitution. Consequently, the gap in rewards-per-quality as well as the difference in agents' costs of effort, contribute to the increasing gap in total rewards for the agents as the degree of substitution increases. When the two algorithms are complements, however, the designer prefers both agents to increase their algorithm quality and sets the same reward per quality to both.[2] Thus, the increasing gap in agents' rewards

[2] Note that the designer does not offer a higher reward per quality to the runner-up compared to the winner.

with an increase in the degree of complementarity arises solely from the difference in agents' costs.

Proposition 2.2

(i) When the algorithms are complements, the ensemble quality is increasing in the degree of complementarity.

(ii) When the algorithms are substitutes, there exists a ω_e such that the ensemble quality is decreasing in the degree of substitution when $\omega > \omega_e$. When $\omega < \omega_e$, there exists a λ_e such that the ensemble quality is decreasing in the degree of substitution if and only if $\lambda \leq \lambda_e$.[3]

Proposition 2.2 examines how algorithmic interdependency affects the ensemble quality, hence value to the designer. The finding that an increase in complementarity increases the ensemble quality is intuitive. The increase in the ensemble quality arises not only from the higher synergy between the two algorithms, *ceteris paribus*, but also from the increase in rewards offered by the designer (Proposition 1(i)) which increases the quality of the individual algorithms. The effect of algorithmic interdependency on the ensemble quality is opposite when the algorithms are substitutes relative to when they are complements except when $\lambda > \lambda_e$ and $\omega < \omega_e$. The negative effect of the degree of substitution on the ensemble quality is intuitive and it can be explained by the adverse impact one algorithm has on the incremental value of the other algorithm when they are substitutes and the designer's general incentive to reduce the rewards (Proposition 1(ii)) in this case. However, an increase in the degree of substitution can also enhance the ensemble quality (Proposition 2(ii)), which is surprising. This counterintuitive finding is explained by Proposition 1(ii) which shows that the designer would prefer to reduce the contribution of the low-quality algorithm and enhance the contribution of the high-quality algorithm with an increase in the degree of substitution if the intrinsic utility is low and the degree of substitution is high. The increase in rewards to the winner induces the low-cost agent to increase his algorithm quality substantially such that this increase offsets the reduction in the algorithm quality of the runner-up, i.e., the high-cost agent.

The analysis in this section suggests that the optimal reward scheme in algorithmic contests could be fundamentally different from the conventional contest rewards scheme – the winner-takes-all scheme. The key differences depend crucially on agents' intrinsic utility and the nature and the degree of interdependency between the algorithms that are ensembled. Notably, we find that the designer should provide the same reward per quality to both the winner and the runner-up when the algorithms are complements. This finding implies that it is optimal for the designer to not choose a winner and a runner-up in algorithmic crowdsourcing contests if the algorithms would be complements. If the algorithms would be substitutes, the designer should pay more to the winner and less to the runner-up, which is consistent with how rewards in conventional contests are structured. Our analysis further reveals that the designer should increase the gap in the rewards between the winner and the runner-up as the degree of substitution increases. Only when the degree of substitution is likely to be significantly large, the designer should resort to the conventional winner-takes-all reward scheme in algorithmic contests.

3. Computational Study of A Real-Life Algorithm Contest

3.1. Classification Algorithms

A classification problem deals with predicting a categorical class that an entity belongs to based on its attributes. For simplicity, suppose there are two possible classes, positive and negative. Algorithms that predict the class based on attributes typically compute a score based on attribute values and classify the entity as belonging to the positive class if the score exceeds a threshold value and to the negative class otherwise. Because of the stochastic nature of the attributes and the relationship between an entity's attributes and the true class, the score follows a statistical distribution and the prediction suffers from false-positive and false-negative errors. The ROC curve of a classification algorithm depicts the true-positive rate as a function of the false-positive rate. Note that the range for both these rates is 0 to 1. The area under the ROC curve, i.e., between the curve and the x-axis, is referred to as the AUC for the classification algorithm. An algorithm that has a higher value for AUC is considered to have superior performance than one that has a smaller AUC.

Let $x_{i|n}$ and $x_{i|p}$ denote the scores computed by algorithm i, $i \in \{1,2\}$, conditional on true negative and true positive classes, respectively. Suppose $x_{i|n}$ and $x_{i|p}$ follow a Gaussian distribution. Specifically, $x_{i|n} \sim \mathcal{N}(\mu_{in}, \sigma_i)$, $x_{i|p} \sim \mathcal{N}(\mu_{ip}, \sigma_i)$ where $\mu_{i.}$ and σ_i respectively denote the mean and standard deviation of algorithm i's score distribution.[4] Then, the AUC of algorithm i is given as (see marzban2004roc for the derivation):

$$AUC_i = \Phi\left(\frac{\mu_{ip} - \mu_{in}}{2\sigma_i}\right). \tag{10}$$

where $\Phi(.)$ denotes the cumulative distribution function of a standard Gaussian distribution. Without loss of generality, we assume that the distributions are normalized such that $\mu_{ip} - \mu_{in} = 1$ for each i and hence

$$AUC_i = \Phi\left(\frac{1}{2\sigma_i}\right). \tag{11}$$

[3] λ_e is the real solution to the equation $16c_1^2 c_2(c_2(\omega+3) - \lambda(\omega+2)) - 4c_1\lambda\left(4c_2^2(\omega+2) - 6c_2\lambda(1+\omega) + \lambda^2\omega\right) + \lambda^3(\lambda(\omega-1) - 4c_2\omega) = 0$. The proof of the proposition shows the existence of ω_e which does not have a closed-form solution.

[4] We assume that the variances of the two distributions, conditional on the true class, are equal for expositional clarity.

Let the quality of an algorithm i be $s_i = 1/\sigma_i^2$. Then, from equation (11), we observe that

$$s_i = 4(\Phi^{-1}(AUC_i))^2, \tag{12}$$

suggesting that there is one-to-one mapping between our quality definition, s_i and the AUC performance of the algorithm, AUC_i, due to the monotonic transformation.

3.2. The Digital Mammography Challenge

To parameterize the model defined in Section 3.1, we use a real-life contest for algorithms: the Dialogue on Reverse Engineering Assessment and Methods (DREAM) initiative's digital mammography crowdsourcing challenge for breast cancer detection. The goal of the digital mammography DREAM challenge was to reduce the rate of false positives without sacrificing the rate of true positives when using mammograms to detect breast cancer, one of the most challenging and costly issues in screening mammography. The DREAM challenge participants were asked to develop and submit algorithms that can predict breast cancer based on mammograms. The challenge organizers promised a cash prize for the winner if the algorithm's performance surpassed that of the radiologists. The organizers assessed the performance of submitted algorithms using AUC.

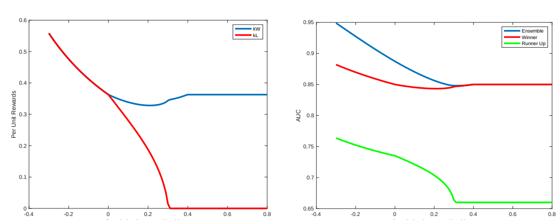
The results of this international challenge were published in Schaffter et al., 2020. One of the coauthors of this paper was part of the organizing team of this challenge and has proprietary access to the predictions by the 18 teams who were part of the final phase. The data from the final submissions form the basis for our computational study.

The AUC of the submissions ranged from 0.52 to 0.87. To be consistent with the theoretical model and to be able to numerically solve the game, we create two point estimates based on high-performing and low-performing submissions for our study. Accordingly, we use the average AUC of the top quartile, 0.85, and the average AUC of the third quartile, 0.70 and use (12) to determine the quality of the winner and runner-up submissions. We compute the conditional correlation coefficients between every pair of those 18 algorithms (i.e., the correlation coefficient between predictions of two algorithms given the true label, positive or negative). The correlation coefficients of the algorithms conditioned on the negative and positive classes were quite similar and ranged -0.2 to 0.7 for λ.

To estimate the effort costs for the participants, we use the hourly wages reported for freelance data scientists in ZipRecruiter, a U.S.-based employment marketplace. Specifically, the site reports an overall average of \$48, \$68 as the first quartile average, and \$25 as the third quartile average.[5] Interpreting the hourly wage as the opportunity cost of a data scientist for participating in the competition, we consider the ratio of the first and third quartiles of these wages as the ratio of costs for the high-cost and low-cost participants, $c_2/c_1 = 2.72$. The organizers of the DREAM challenge did not take into account the possibility of ensembling the submitted algorithm when designing the rewards structure, nor did they announce to participants that they would form an ensemble. Substituting the point estimates for the algorithm qualities and the ratio of costs into the equilibrium quality expressions presented in Lemma 2.1 for the contest where the ensemble is infeasible, we obtain an estimate for the participants' intrinsic utility in the contest. The computation results in a point estimate of $\omega = 0.27$. Using the estimated parameters, we solve the game in (4) for different values of λ to characterize the optimal rewards and contest outcomes.

Figure 1 shows the optimal reward per unit quality for the winner and the runner-up for different correlation coefficients, assuming that the designer would ensemble the algorithms.[6] Consistent with our theoretical results, we find that the optimal reward per unit quality for the low-cost agent, k_W, first decreases and then increases to a fixed value, as depicted in Figure 1(a). However, the reward per unit quality for the high-cost agent, k_L, is non-increasing with correlation, and becomes zero when $\lambda \geq 0.31$. Thus, if $\lambda \geq 0.31$, it is optimal to adopt the conventional winner-takes-all reward scheme in the DREAM challenge. Figure 1(b) shows the AUC of the two algorithms and the ensemble AUC under the optimal reward scheme. The ensemble AUC follows a pattern similar to that of the reward for the winner; the ensemble AUC first decreases and then increases in the correlation coefficient, finally staying constant beyond a correlation coefficient threshold. The AUC of the ensemble ranges from 0.848 to 0.927 for the range of correlation coefficients observed in the DREAM challenge. The AUC of the winner's algorithm and that of the runner-up's range from 0.843 to 0.869 and from 0.66 to 0.752 respectively. The winner's and the runner-up's AUC ranges roughly correspond, respectively, to those in the first and third quartiles in the DREAM challenge. Figure 1 demonstrates that the theoretical results of Section 4 hold for classification tasks with generally accepted algorithm quality and interdependency measures.

Figure 1: Rewards and Performance for the DREAM Challenge



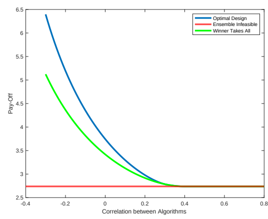
(a) Rewards (b) Quality

[5] https://www.ziprecruiter.com/Salaries/Freelance-Data-Scientist-Salary, accessed on July 20, 2021.

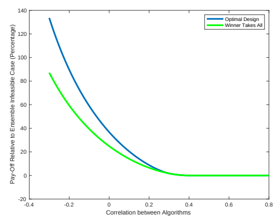
[6] Recall that the computational analysis was carried out under the assumption that the designer's value per unit quality is normalized to unity. Therefore, the reward and payoff values should be interpreted accordingly.

Next, we assess the value of using the optimal rewards scheme along with ensembling, relative to the other two benchmarks: winner-takes-all scheme with ensembling and winner-takes-all without ensembling. We assess the value in terms of the designer's payoff as well as the final algorithm quality, and it is shown in Figure 2.

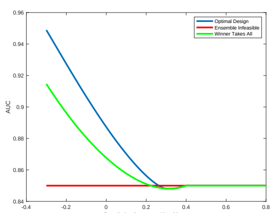
Figure 2: Value of the Optimal Reward Design in DREAM Challenge



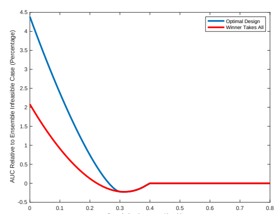
(a) Absolute Designer Payoff



(b) Payoff Gain as a Percentage Relative to when Ensemble is Infeasible



(c) Absolute AUC



(d) AUC Gain as a Percentage Relative to when Ensemble is Infeasible

In Figure 2(a), we observe that when algorithms are complements ($\lambda < 0$) or the degree of substitution is low ($0 \le \lambda < 0.31$), the optimal design offers more payoff than both benchmark designs. The percentage payoff gain from the optimal design as well as from ensembling but with winner-takes-all reward scheme relative to the ensemble infeasible case diminishes as the correlation between the two algorithms increases, as depicted in Figure 2(b). When the degree of substitution is high, i.e., $\lambda \geq 0.4$, all three designs offer the same pay-off to the designer. In Figure 2(c), we observe a slightly different pattern in terms of AUC performances. When algorithms are complements, $\lambda < 0$, the AUCs are decreasing in λ with the optimal design offering the highest AUC performance, followed by the case when ensembling is done under the winner-takes-all scheme and then by the the conventional design in which the ensemble is infeasible; the relative gain also diminishes as λ increases. However, when algorithms are substitutes, $\lambda \geq 0$, the ensemble performance of the design without ensembling can be larger than those with ensembling if $\lambda < 0.4$. For large correlation between the algorithms, $\lambda \geq 0.4$, the AUC performance of the ensemble algorithms is the same in all three designs.

Together, these numerical findings confirm our theoretical insights on how to design rewards for contests for algorithms and how their design and the outcomes differ from conventional contests. Moreover, the results show that the optimal reward design that takes into account the unique characteristics of algorithm contests such as ensembling, algorithm correlation, and participants' intrinsic motivation can lead to substantial payoff and quality gains when compared to the conventional winner-takes-all reward scheme.

4. Discussion and Conclusion

This paper examines how the unique features of prediction algorithms affect the reward design for algorithmic crowdsourcing contests. In particular, we show how the designer's ability to create an ensemble of algorithms affects the structure of rewards that induce participants to exert algorithm development effort. The value from the ensemble depends not only on the individual quality of the algorithms that are ensembled but also on the interdependency (i.e., correlation) between them. We demonstrate that the correlation between the submitted algorithms determines whether they are substitutes or complements to each other in improving the ensemble quality and hence the optimal reward structure for the contest. If the algorithms are expected to be complements, it is optimal for the designer to use a reward scheme that does not distinguish between a winner and runner-up (with respect to reward per unit quality), but if the algorithms are expected to be substitutes, then it is optimal for the designer to distinguish between the winner and the runner-up by using a larger reward per unit quality to each. The rewarding of both the winner and runner-up under complementarity or low-substitutability is in contrast to the conventional reward scheme, the winner-takes-all, that is used in contests where ensembling solutions are not feasible. When the algorithms are highly substitutable, however, it is optimal for the designer to use the conventional winner-takes-all reward scheme. We further show that the contest participants' intrinsic utility from participation, which is an important feature in innovation contests such as the one we examine, plays a critical role in the optimal reward scheme as well. We use a real-life algorithmic crowdsourcing competition in the mammography-based breast cancer prediction context to demonstrate that our theoretical insights hold for this contest and that the value of using the optimal reward scheme relative to conventional approaches can be significant.

Our findings provide valuable insights to contest designers, technology platforms that enable and encourage crowdsourcing contests, academic scholars, and stakeholders that are decidedly concerned about contest outcomes such as those that rely on algorithms to make decisions. While we highlight the importance of the need for designers to understand the nature of contributors' inputs and hence its role in the reward design, there are important takeaways for platform designers as well. For instance, platforms that host algorithmic contests, such as Kaggle, should offer features that facilitate the participation of agents with diverse skills so that their contributions have little overlap (i.e., more independence) in terms of the solution approach. Intervention mechanisms such as leader boards, descriptive summaries or documentation of participant contributions at each stage, feedback mechanisms for revealing participant strengths (e.g., profile pages, links to public works highlighting participant capabilities) can all help in this regard. An indirect benefit of mechanisms that encourage independent algorithms is that designers would not be inclined to choose winners and reward multiple contributors equally, which could also encourage more participation.

From an academic standpoint, our model setup and findings highlight an increasingly relevant problem context which prior scholars have not focused on. Contests for algorithms depart from the work in the prior literature mainly because of two reasons: (i) Participant behavior is affected by how the interdependency between different submissions shapes the final solution for the problem posed by the designer. Our approach in modeling the marginal value of one algorithm with respect to the value offered by the other algorithms in the ensemble captures the resulting idiosyncracies in participant behavior (i.e., algorithmic effort). (ii) Although some prior work has modeled the objective of the designer as a combination of solutions, our work extends the existing literature by explicitly modeling the mechanics of how an ensemble is obtained for predictive algorithms. Overall, the distinctive nature of algorithms implies that existing theories of the contest and tournament design should be reexamined in this context to extend these theories and generate more nuanced insights.

References

Adjerid, I., Ayvaci, M. U. S., & Özer, Ö. (2023). Value of algorithm-enabled process innovation: The case of sepsis. *Manufacturing & Service Operations Management*.

Adomavicius, G., Bockstedt, J. C., Curley, S. P., & Zhang, J. (2018). Effects of online recommendations on consumers' willingness to pay. *Information Systems Research*, *29*(1), 84–102.

Ahsen, M. E., Ayvaci, M. U. S., & Mookerjee, R. (2023). When machines will take over? algorithms for human-machine collaborative decision making in healthcare. *56th Annual Hawaii International Conference on System Sciences, HICSS 2023*, 5733–5740.

Ahsen, M. E., Ayvaci, M. U. S., & Raghunathan, S. (2019). When algorithmic predictions use human-generated data: A bias-aware classification algorithm for breast cancer diagnosis. *Information Systems Research*, *30*(1), 97–116.

Ahsen, M. E., Vogel, E. M., & Stolovitzky, G. A. (2019). Unsupervised evaluation and weighted aggregation of ranked classification predictions. *Journal of Machine Learning Research*, *20*(166), 1–40.

Chesbrough, H., Vanhaverbeke, W., & West, J. (2006). *Open innovation: Researching a new paradigm*. Oxford University Press on Demand.

Cohen, C., Kaplan, T. R., & Sela, A. (2008). Optimal rewards in contests. *The RAND Journal of Economics*, *39*(2), 434–451.

Halac, M., Kartik, N., & Liu, Q. (2017). Contests for experimentation. *Journal of Political Economy*, *125*(5), 1523–1569.

Iansiti, M., & Lakhani, K. R. (2020). From disruption to collision: The new competitive dynamics. *MIT Sloan Management Review*, *61*(3), 34–39.

Jeppesen, L. B., & Lakhani, K. R. (2010). Marginality and problem-solving effectiveness in broadcast search. *Organization Science*, *21*(5), 1016–1033.

Kadiyala, B., Özer, Ö., & Şimşek, A. S. (2021). Data-driven approaches to targeting promotion e-mails: The case of delayed incentives. *Production and Operations Management*, *30*(3), 766–782.

Kaufmann, N., Schulze, T., & Veit, D. (2011). More than fun and money: Worker motivation in crowdsourcing-a study on mechanical turk. *Proceedings of the 2011 AMCIS, Paper 340 (12 pages).*

Körpeoğlu, E., & Cho, S. (2018). Incentives in contests with heterogeneous solvers. *Management Science*, *64*(6), 2709–2715.

Moldovanu, B., & Sela, A. (2001). The optimal allocation of prizes in contests. *American Economic Review*, *91*(3), 542–558.

Schaffter, T., Buist, D. S. M., Lee, C. I., Nikulin, Y., Ribli, D., Guan, Y., Lotter, W., Jie, Z., Du, H., Wang, S., et al. (2020). Evaluation of combined artificial intelligence and radiologist assessment to interpret screening mammograms. *JAMA Network Open*, *3*(3), e200265–e200265.

Song, Y., Sahoo, N., & Ofek, E. (2019). When and how to diversify—a multicategory utility model for personalized content recommendation. *Management Science*, *65*(8), 3737–3757.

Terwiesch, C., & Xu, Y. (2008). Innovation contests, open innovation, and multiagent problem solving. *Management Science*, *54*(9), 1529–1543.