Mechanical TA 2:
Peer Grading With TA and Algorithmic Support
(A preprint)

Hedayat Zarkoob
University of British Columbia
Canada
hzarkoob@cs.ubc.ca

Kevin Leyton-Brown
University of British Columbia
Canada
kevinlb@cs.ubc.ca

ABSTRACT

Mechanical TA 2 (MTA2) is an open-source, distributed peer grading system that boosts performance by leveraging both trusted TAs and computationally intensive algorithms. The system provides a unified platform for submission of assignments, grading by both peers and TAs, and reporting of feedback. It also supports dividing students into different pools based on their peer-grading prowess; mechanisms for automated calibration and spot checking; and the ability for students to appeal grades and to give feedback about individual reviews. Bayesian inference and mixed-integer programming algorithms perform interpretable aggregation of peer grades and estimate students' grading performance, providing feedback, incentivizing high-quality grading, and directing TA spot checks appropriately. Analysis of data from four offerings of a large undergraduate class provides empirical evidence of MTA2's effectiveness.

CCS CONCEPTS

• **Applied computing** → **Collaborative learning**.

KEYWORDS

Peer Grading, Educational technology, Crowdsourcing

ACM Reference Format:
Hedayat Zarkoob and Kevin Leyton-Brown. 2023. Mechanical TA 2: Peer Grading With TA and Algorithmic Support (A preprint). In *The Technical Symposium on Computer Science Education, March 20–23 2024, Oregon.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Peer grading can improve students' learning by encouraging critical thinking about the work of their classmates and can also help with the delivery of large classes by reducing grading work for teaching assistants (TAs) and instructors [19]. A wide variety of different peer grading systems already exist [2, 4, 7, 14, 16]. All of these systems either work without relying on TAs or use them only at a very basic level (e.g., resolving student appeals). Orthogonally, none of these systems makes use of computationally intensive AI techniques (such as probabilistic inference or discrete optimization). This paper argues that the integration of both TAs and such AI algorithms can help to overcome three key drawbacks with current peer grading systems, which we dub incentives, standardization, and scaling.

Incentives. Peer grading systems only work when students take their grading tasks seriously. While some students behave altruistically or recognize that peer grading helps their own learning, others are motivated by more immediate incentives. Many peer grading systems allow students to appeal when they disagree with a peer review [4, 16]; some systems incorporate TAs to review grades that are notably high [23]. These approaches respectively discourage reviewers from assigning overly low or high grades. Some systems further reward students for choosing grades that are similar to those assigned by other peers to the same submissions [4, 8]. Unfortunately, neither of these mechanisms discourages students from giving close-to-average grades to every single submission [5, 6], offering students a low-effort alternative to grading sincerely. TAs can partially address this issue by "spot checking" a randomly chosen fraction of submissions [6, 9, 23, 25]. When a submission is spot checked, a TA grades the submission carefully, evaluates the reviews given to that submission, and rewards or punishes the graders. At sufficient levels, spot checking offers students strong incentives to grade sincerely [5, 6], but can be very expensive. It is thus appealing to supplement spot checking with more direct methods for detecting (likely) insincere or low-effort grading, such as probabilistic inference [24].

Standardization. Unless peer grading systems teach students to grade consistently and accurately, different students are likely to assign different grades to the same work [2, 12, 21, 23]. There are two key ways that TAs can help in this task. The first is by preparing so-called *calibration submissions*, such as submissions from previous years, which can then be assigned to all students in the class [2]. Immediately after a student grades a calibration submission, they can be shown the TA grade and given its rationale; such instantaneous feedback has been shown to aid learning [11]. The second is to let TAs give personalized feedback to students on their grading performance. Calibration can work with very limited TA resources, but of course it is also valuable to have TAs provide direct feedback on actual peer grading (triggered, e.g., by a student appeal, a random spot check, or some other signal such as a pattern of suspiciously similar grades). Probabilistic inference algorithms can also contribute to this latter task by computing estimates of students' grading prowess based on their patterns of agreement and disagreement with their peers, offering students a second means of receiving feedback [13, 17, 22].

Scaling. In order for a peer grading system to scale, it must assign aggregated grades in a way that benefits from the presence of reliable students who need little supervision, without being derailed by

SIGSCE '24, March 20–23, 2024, Portland, Oregon

ACM ISBN 978-1-4503-9999-9/18/06…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

others who are less skilled or more erratic. Most current systems aggregate grades as either the mean or median of reports, making them unable to take grader reliabilities into account. Systems that grade students for their peer grading reliability must furthermore estimate *these* grades, again presenting a scalability challenge. Some methods propose using "point estimates" for such values [1, 3, 4, 8, 18, 20]. However, these approaches can lead to inconsistent or excessively confident results [24]. Instead, probabilistic inference can be used to compute joint distributions over both grades and grader reliability estimates [13, 17, 22], though only one recent paper [24] has shown how these techniques can be combined with incentives for honest reporting. To date none of these probabilistic inference methods has been deployed in a practical peer grading system.

To our knowledge, the most commonly used peer grading applications are PeerScholar (PS) [14], Peer Grader (PGR) [7], Calibrated Peer Review (CPR) [2], Crowdgrader (CG) [4], Peergrade (PG) [16], Kritik [10], and Peerceptiv (PC) [15]. Table 1 compares each application alongside MTA1 and MTA2. Space does not permit us to discuss each system, but it is important to explain how MTA2 differs from MTA1. MTA1 was used in a computer science course called "Computers and Society" for several years. A study published at SIG-CSE [23] demonstrated the system's usefulness: specifically, MTA1 significantly reduced the need for TA labor in a large class and its use of spot checking, calibration, and division of students into multiple pools helped students to grade more accurately and to learn the course material. However, MTA1 was highly specialized to a single and many of its design choices were hard coded, making the system difficult to change. MTA1 also lacked algorithmic support and was not suitable for use at scale.

MTA2 is a complete reimplementation of MTA1 (in Python rather than PHP) that improves performance, reliability, and extensibility and also introduces various new features. MTA2 has two main parts: a web front end and a computation back end. The front end allows students to submit their assignments and receive feedback, and has a modular design that makes customization easy. It also supports dividing students into different pools based on their peer-grading prowess; mechanisms for automated calibration and spot checking; and the ability for students to appeal grades and to give feedback about individual reviews. On the back end, an inference engine uses probabilistic reasoning and mixed integer programming to perform interpretable aggregation of student-reported grades and to estimate students' grading performance. The system uses these estimates to offer feedback to students, to incentivize high-quality grading, and to direct TA spot checks to where they will do the most good.

The design of MTA2 is modular, allowing instructors to choose features appropriate for their courses and making it easy to change system logic. The design of peer grading mechanisms is an active area of academic research; MTA2's modular design also makes it a good environment for evaluation (e.g., of different spot checking or grade aggregation mechanisms). Beyond its architectural advantages, MTA2 also offers four new system-level features. First, MTA2 allows student assignments to consist not only of ASCII text but also PDFs and answers to multiple choice questions, rendering each directly in the browser. Second, students can flag inappropriate reviews independently of the appeal process, which they may not be motivated to use when an offending review did not impact their grade. In our early deployment of MTA2, we found that this feature led to a considerable increase in student satisfaction. Third, MTA2 allows instructors to upload sets of student submissions as a single zip file. This enables the ingestion of exams and quizzes conducted outside of MTA and optionally permits them to be peer graded. Finally, MTA2 offers advanced SAML2 integration, allowing users to log in using organizational accounts (e.g., university-wide login systems), increasing ease of use, security, and regulatory compliance.

The rest of this paper is organized as follows. Section 2 introduces the main features of MTA2, summarizing the user interface, front-end features, and back-end features. Section 3 describes deployment of MTA2 and Section 4 concludes.

2 MTA2: KEY FEATURES

Here we discuss in turn MTA2's user interface, its front-end features, and its back-end features.

2.1 User interface

When a user logs into MTA2, they see a list of the courses they are associated with and the role they have in each. Users can associate themselves with a new course by entering a given code. Upon choosing a course, every user sees an interface divided into four main tabs: dashboard, assignment, review, and appeal.

2.1.1 Dashboard. The dashboard is a one-screen overview shown at login; for instructors, it shows assigned tasks, a summary of course statistics, and a link to a list of all students (see Figure 1 on the left). By clicking on a student's name in this list, TAs and instructors can see every task that the chosen student has done throughout the course and the corresponding grades they were given. Finally, the course configuration (course name, visibility to students, enrollment codes, etc) can be viewed; which can only be edited by the instructors.

For students, the dashboard contains two main tables, as illustrated in Figure 1 on the right. The first shows assigned tasks and deadlines. The second shows grades for previous assignments. Students can view their submissions and their reviews in detail by clicking on the associated grade. They can also appeal or flag a review if necessary. Student reviews are only available after the student review deadline is passed. The final score of each submission is shown once the TA review deadline is reached. (Observe that TAs might spot check an assignment and thereby change the assigned grade.) When multiple pools are enabled, the dashboard also indicates whether the student is supervised or independent.

2.1.2 The Assignment tab. Instructors and TAs can see the list of all assignments and create new ones using this tab. They can also view the submissions for each assignment by displaying the submission list. Instructors can furthermore determine whether an assignment is visible to students and set deadlines.

For students, the Assignment tab shows students the list of their assignments; for assignments that are not yet due, students can edit their submissions. Students can also see and appeal their grades in this tab once the deadline for TA reviews has passed.

2.1.3 The Reviews tab. For students, this tab displays two tables: pending reviews and previously submitted reviews. Students can still edit submitted reviews until the review deadline. A student's assigned reviewing tasks include both peer evaluation of other students' work and calibrations to (self/) assess reviewing ability. Reviewing an

Feature		PS	PG	CPR	CG	PGR	Kritik	PC	MTA1	MTA2
Incentives via	TA spot checks	–	–	–	–	–	–	–	✓	✓
	grade consensus	–	–	✓	✓	✓	–	✓	–	✓
	review the reviewer	✓	✓	–	–	✓	–	–	–	✓
Standardization via	calibration submissions	–	–	✓	–	–	✓	–	✓	✓
	instructor/TA feedback	–	✓	✓	✓	✓	✓	✓	–	✓
	algorithmic feedback	–	✓	–	✓	–	–	✓	–	✓
	dividing graders into pools	–	–	–	–	–	–	–	✓	✓
Scaling via	auto-weighting peer grades	–	–	–	✓	–	–	✓	–	✓
	auto-grading peer graders	–	–	–	✓	–	–	✓	–	✓
Instructors can enable/disable features		✓	–	✓	–	✓	–	–	–	✓
Instructors can change system logic		–	–	–	–	–	–	–	–	✓
Support for different assignment types		✓	✓	–	✓	✓	✓	✓	–	✓
Students can give feedback about individual reviews		–	–	–	–	✓	✓	–	–	✓
Provide data-driven insights to users		–	✓	–	–	✓	✓	✓	–	✓
Students can review themselves (self-assessment)		–	✓	✓	–	✓	✓	✓	–	✓
Instructors can choose to grade individual submissions		–	–	–	✓	✓	✓	✓	–	✓
Students and instructors can discuss submissions		✓	✓	–	✓	–	✓	–	–	–
Students can hand in group assignments		–	–	–	✓	✓	✓	✓	–	–
Support for checking submission similarity		–	–	–	✓	–	–	–	–	–

Table 1: Comparing peer grading applications across both our three desiderata and 10 other features across which they differ.

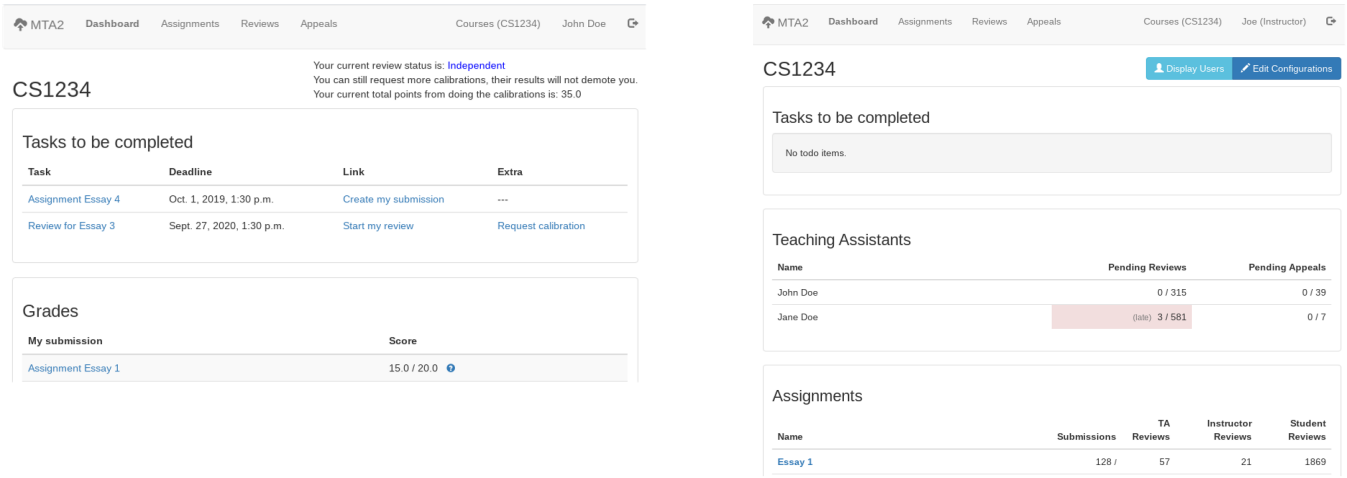

Figure 1: The student dashboard (left) and instructor dashboard (right).

assignment means answering multiple-choice rubric questions that describe different elements of an assignment. Each rubric question can also have a reasoning field, which asks the grader to provide a rationale for their choice. The rationale field can be set to require entries between a minimum and maximum length. Submitted reviews can later be evaluated by TAs (using a different rubric). Instructors and TAs can assign reviews and evaluations, create rubrics, and see review stats from the review tab. If an assignment has more than one question, it is possible to assign different TAs to grade different questions. The algorithm used to assign TAs to assignments or to assign spot checks can be changed straightforwardly.

Instructors and TAs can also access the list of submitted reviews and view each one from this tab. When viewing a student review, instructors and TAs can request an evaluation for the review. Note that review evaluation is most naturally part of the spot-checking process. However, if instructors or TAs encounter a review that requires evaluation (e.g., via an appeal), they can also do so directly.

2.1.4 The Appeals tab. The appeals tab shows each TA a list of assigned appeals along with their current status. They can also

reassign an appeal that they cannot resolve themselves. Instructors can see all the appeals. For students, the appeals tab keeps track of all previously submitted appeals and their current status. New appeals are submitted via the assignment tab.

2.2 Front-End features

MTA2 contains various features which instructors may turn on or off as appropriate for their teaching needs. This section describes four features which instructors may easily toggle: spot checking, allowing students to offer feedback about individual reviews; calibration; and supervised and independent pools. For users requiring finer grained customization, the MTA2 codebase is written in Python, which makes modifying its behavior simple. The source code is divided into eight directories based on function: assignment, review, evaluation, calibration, grade, account, course, and homepage. The contents of each of these directories is further decomposed into three parts: *templates and views*, which control the UI; *base*, which contains the logic; and *models*, which contain the definitions of data types. Hence a newcomer attempting to modify a specific behavior can quickly identify the part of source code that requires modification.

2.2.1 Spot checking. In MTA2, a spot check has two parts. The first is a *TA review* of the assignment: the chosen assignment is graded by a TA, using the same grading rubric used by students. Second is *TA evaluation* of the peer graders: the TA uses a separate rubric to evaluate the reviews submitted by students who graded the spot checked assignment. By default, the system lets the instructor choose the number of assignments that need to get spot checked and assigns TAs to students uniformly at random. However, this logic can be revised easily, for example targeting high grades or by using the inference output. Furthermore, TAs are free to browse through assignments and conduct additional spot checks, based e.g. on reviewer identity, degree of agreement between reviewers, or peer grade assigned.

2.2.2 Flagging individual reviews. MTA2 also allows students to (both negatively and positively) flag individual reviews. The procedure is similar to appealing, but it is supported via a separate mechanism for two reasons. First, we found it useful to allow students to identify rude, unfair, or low quality reviews even when these reviews do not affect the student's overall grade. If a student believes that a review is inappropriate, they can flag the review and provide their reasoning. The report is then assigned to a TA as in an appeal. If the TA confirms that the report is inappropriate it no longer affects the assignment's grade. The reviewer is also notified and the TA may take additional action outside the MTA2 system. Second, we wanted to allow students to give positive feedback about particularly good reviews, to help reward students for making exceptional efforts.

2.2.3 Calibration submissions. Calibration submissions have known "ground truth" grades; for example, they might be carefully graded submissions from previous offerings of a course. They are graded using the same rubric as student submissions and can either be presented separately to students in the UI (to support a workflow like requiring a calibration review before peer reviews) or mixed together with student submissions (to drive TA spot checks by detecting students who grade poorly). MTA2 auto-grades calibration reviews and shows students the correct answers right after they have submitted their review.

2.2.4 Supervised/independent pools. MTA is able to maintain different *pools* of students, where students are only assigned peer reviews for others in the same pool. The first, "supervised" pool guarantees that every submission and review submitted by students will also be graded by a TA, with only the TA grade counting for evaluation. The purpose of this pool is to ensure that students do not assign binding peer grades until they have shown themselves to be fair and consistent graders. At the beginning of the course, all students are assigned to the supervised pool. They graduate to the second, "independent" pool by demonstrating consistent performance either in calibration reviewing or in real peer review in the supervised pool. Students in the independent pool are evaluated for the quality of their grading only if they are spot checked, the submission they have graded is appealed, or their review is flagged as inappropriate.

The use of supervised and independent pools offers a second benefit for calibration: they dramatically reduce TA grading when a new course starts. Performing calibration reviews can help supervised students to become independent more quickly—often, even before submitting the first assignment. Instructors can also require supervised students to perform additional, weekly calibration reviews, both to reduce TA workload and to help weaker students to learn to peer grade effectively.

2.3 Back-End features

MTA2's inference engine employs the probabilistic model outlined in [24]. This model takes reported grades from students and TAs as inputs and generates estimates for the actual submission grades. It also provides estimations for the effort put in (calculated as 1 minus the probability of choosing a grade close to the class average without considering the essay) and a reliability parameter estimating a student's tendency to differ from the essay's true grade when making an effort.

2.3.1 Dependability scores. The dependability score is an estimate of a student's effort probability multiplied by their reliability. The system starts out with the assumption that all students have low dependability scores (specifically, low effort and high variance). As students grade assignments and perform calibrations, the system will update these beliefs. In particular, observe that doing more calibrations both helps the students to get better at grading and gives evidence of students' grading prowess to overwhelm the system's pessimistic initial belief. Note that if students always assign each submission the class average, or if they grade very erratically, the model will assign them a low effort probability; they need to properly identify both strong and weak assignments in order to achieve a high dependability score. For each student, MTA2 maintains both a *realistic* (mean) estimate of dependability and a *pessimistic* (lower confidence bound) estimate. Below we describe how MTA2's frontend can use these estimates to improve its performance.

2.3.2 Identifying supervised students. Each week, MTA2 will identify those students with pessimistic dependability estimates below a certain threshold and assign them to the supervised pool (Note

that this means that if a student has a low pessimistic dependability estimate, they will likely get noisier peer grades, but this won't matter because their grade will be entirely determined by a TA.) The remainder of students (those for whom even our pessimistic dependability estimates exceed a threshold) will perform independent peer grading. This mechanism ensures that a student's assignment will either be graded by a TA or will be graded entirely by peers that the system confidently predicts will grade reliably.

2.3.3 Evaluating students grading performance. Each week, MTA2 will assign students a peer grading score; overall, these scores can make up the peer grading portion of students' final grades. These grades will be derived from the realistic estimates of their dependability score. Each instructor has the flexibility to choose the exact mapping from dependability student to actual grades. For example, they can set the mapping such that a student with a dependability score exactly equal to the calibration threshold would receive a grade of 80%. Beyond this, the formula can strictly monotone in dependability.

2.3.4 Interpretable grades. When no TA grades a submission, its grade will be computed by the inference engine using a mixed integer program which outputs submission grades as weighted averages of the grades assigned by peers, where each peer's weight is proportional to MTA2's assessment of their dependability. In cases where MTA2 considers a peer grade unreliable, it may assign it a weight of zero.

3 DEPLOYING MTA2

To date, MTA2 has been deployed in four iterations of an in-person undergraduate-level computer science course centered around computers and society from the Fall of 2019 to 2022. These iterations accommodated between 118 and 194 students, each encompassing 11 assignments. The majority of student essays were assessed by 4–5 peers, while a few received more or fewer grades. Every essay was graded on a scale of 0–20, divided into four components (structure, evidence, subject matter, English), each graded between 0 and 5. A team of 3–5 TAs supported each course, conducting spot checks on 474–934 essays. Their focus was primarily on essays with suspiciously high average grades, substantial disparities among peer grades, and graders in the supervised pool. In the initial two iterations, MTA2 operated solely with TA support. We added probabilistic inference in the final two iterations.

For our analysis, we employed the MTA2 inference engine to estimate students' true grades and dependabilities across all four years. We adopted a similar methodology as described in [24] to determine an appropriate set of hyperparameters for our datasets; anyone applying MTA2 in their own course should likewise fine tune these hyperparameters. We also conducted end-of-year surveys to gather feedback for improving the next course offering. Although this survey was not initially intended as a research tool, we report two categories of responses which we found particularly useful for assessing students' experiences with MTA2.

3.1 Learning outcomes

3.1.1 Do students get better at grading? As described in Section 2.3.3, dependability scores can serve as a proxy for evaluating students' grading performance. Figure 2 illustrates that students' dependabilities improved by as much as 36%, notably during Fall 22, which is the year when we fully leveraged the capabilities of the inference engine. This evidence strongly suggests that MTA2 is effective in teaching students to assess each other's work.

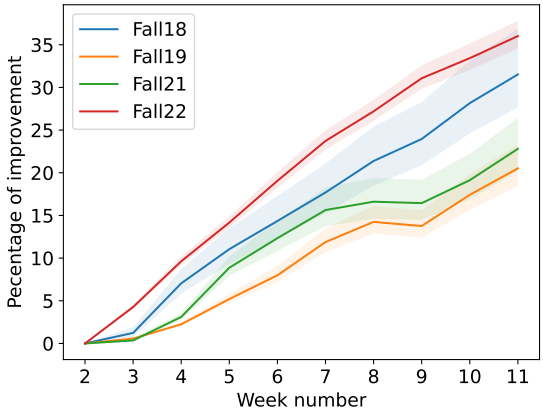

Figure 2: Student dependabilities calculated by MTA2 inference

3.1.2 Do students get better at writing essays? We observed a gradual improvement in student grades over time. Since both TAs and peers were calibrated to a static scale using objective evaluation criteria, this indicated that students were indeed learning to write better. However, this improvement cannot be attributed simply to MTA2; e.g., they might have improved because of attending lectures. To focus in on MTA2's impact, we examine the 20% of students with respectively the highest and lowest dependability scores in the last week of the course—i.e., those students who were worst and best at peer grading—to see if their distribution of essay grades differed. We observed that in Fall 18, 19, and 22, the empirical distribution of grades for the top 20% of graders stochastically dominated (i.e., lay to the right of) the empirical distribution of grades for the worst 20% of graders. We illustrate the distributions for Fall 22 in Figure 3; Fall 18 and 19 data were qualitatively similar. Fall 21 differed; while fewer good graders received very low grades, they also less often received very high grades (see Figure 4).

3.2 Student surveys and observational data

3.2.1 Perceived fairness of grades. In our annual end-of-year survey, we asked students whether they found the aggregated peer grades assigned by MTA2 to be fair. We observed an increase in the average score from 2.7 in Fall 18 and 19 to 3.25 (out of 5) in Fall 21 and 22, coinciding with the introduction of the inference engine. While the median remained constant, the first quantile rose from 2 to 3, indicating a reduction in the number of students who perceived their grades to be highly unfair (see Figure 5).

3.2.2 Reporting individual bad reviews. We examined the frequency of appeals submitted by students each year. We observed an initial rate of around 1 appeal per student in Fall 18, which subsequently decreased to 0.65 in Fall 19, and further reduced to 0.57 in Fall 21 and 22. While many factors may have contributed to this

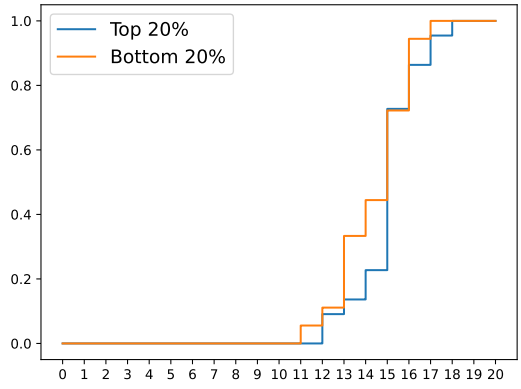

Figure 3: ECDF of Students' Essay 11 grades in Fall 22

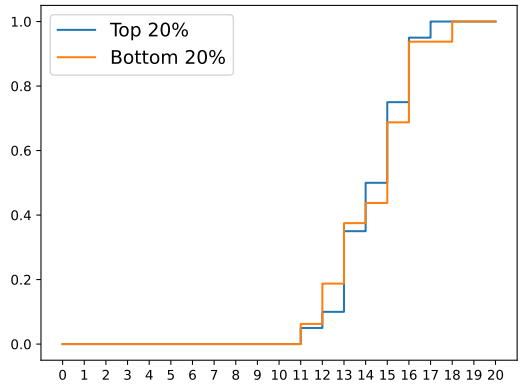

Figure 4: ECDF of Students' Essay 11 grades in Fall 21

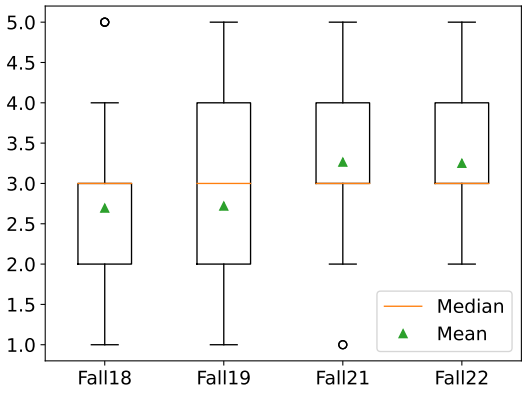

Figure 5: Student perceptions of fairness of MTA2 grades

reduction, our qualitative experiences in class led us to attribute a significant portion of the decline to the introduction of the option for students to report individual problematic reviews (instead of appealing) starting in Fall 19.

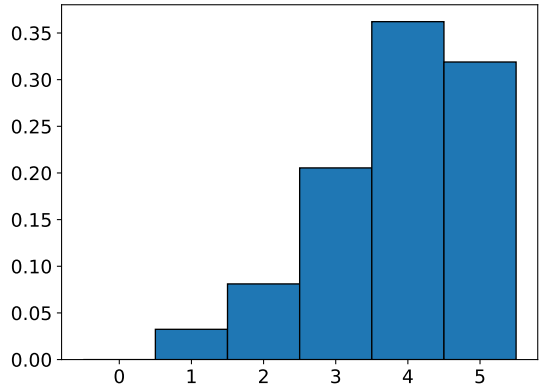

Figure 6: MTA2 user interface ranking

3.2.3 User interface. In Fall 2019, 21, and 22, we asked students how user friendly they found the MTA2 interface. Overall, 68% scored the user interface either 4 or 5 out of 5 (see Figure 6). We contrast this with MTA1; while we are aware of no comparable survey data about that system, its user interface was the subject of many complaints both oral and written, which indeed served as one of the motivations for MTA2's system redesign.

4 CONCLUSION

Mechanical TA 2 (MTA2) is an open-source, web-based system that facilitates peer grading with TA support. It constitutes an advance over existing, widely used peer grading systems via improved incentives for students to invest effort in grading well, mechanisms for standardizing grades across students and TAs, and scaling to large classes. The system's unified platform provides an environment for assignment submission and grading by both peers and TAs. MTA2 supports separating reliable and inconsistent peer graders into separate pools; standardizing graders via calibration; conducting TA spot checks; grade appeals and flagging problematic reviews. The system leverages probabilistic inference to identify strong and weak graders and to infer grades for each submission and uses mixed-integer programming to rationalize inferred grades as interpretable weighted sums of given peer grades. We deployed the system in four offerings of a large class, and found that it improved both student learning and students' happiness with the course. In future work we intend to continue studying the system's educational impact and streamlining its design. For example, the front-end to back-end interface currently requires a partially manual process of exporting grades, running the inference on a separate machine, and then re-importing the results, which we aim to better automate.

REFERENCES

[1] A. Chakraborty, J. Jindal, and S. Nath. 2018. Incentivizing effort and precision in peer grading. *arXiv preprint arXiv:1807.11657* (2018).

[2] O. L. Chapman. 2001. Calibrated Peer Review. http://cpr.molsci.ucla.edu.

[3] K. Cho and C. D. Schunn. 2007. Scaffolded writing and rewriting in the discipline: A web-based reciprocal peer review system. *Computers & Education* 48, 3 (2007), 409–426.

[4] L. De Alfaro and M. Shavlovsky. 2014. CrowdGrader: A tool for crowdsourcing the evaluation of homework assignments. In *Proceedings of the 45th ACM Technical Symposium on Computer Science Education*. ACM, 415–420.

[5] L. De Alfaro, M. Shavlovsky, and V. Polychronopoulos. 2016. Incentives for truthful peer grading. *arXiv preprint arXiv:1604.03178* (2016).

[6] A. Gao, J. R. Wright, and K. Leyton-Brown. 2019. Incentivizing evaluation with peer prediction and limited access to ground truth. *Artificial Intelligence* 275 (2019), 618–638.

[7] E. F. Gehringer. 2001. Electronic peer review and peer grading in computer-science courses. *ACM SIGCSE Bulletin* 33, 1 (2001), 139–143.

[8] J. Hamer, K. T. Ma, and H. H. Kwong. 2005. A method of automatic grade calibration in peer assessment. In *Proceedings of the 7th Australasian Conference on Computing Education-Volume 42*. Australian Computer Society, Inc., 67–72.

[9] R. Jurca and B. Faltings. 2005. Enforcing truthful strategies in incentive compatible reputation mechanisms. In *International Workshop on Internet and Network Economics*. Springer, 268–277.

[10] KritikEducationInc. [n. d.]. Kritik. https://www.kritik.io/. Accessed: 2021-01-17.

[11] J. A. Kulik and C. C. Kulik. 1988. Timing of feedback and verbal learning. *Review of Educational Research* 58, 1 (1988), 79–97.

[12] C. Kulkarni, K. P. Wei, H. Le, D. Chia, K. Papadopoulos, J. Cheng, D. Koller, and S. R. Klemmer. 2013. Peer and self assessment in massive online classes. *ACM Transactions on Computer-Human Interaction (TOCHI)* 20, 6 (2013), 33.

[13] F. Mi and D. Yeung. 2015. Probabilistic graphical models for boosting cardinal and ordinal peer grading in MOOCs.. In *Proceedings of the 29th AAAI Conference on Artificial Intelligence*. 454–460.

[14] D. E. Paré and S. Joordens. 2008. Peering into large lectures: examining peer and expert mark agreement using PeerScholar, an online peer assessment tool. *Journal of Computer Assisted Learning* 24, 6 (2008), 526–540.

[15] Peerceptiv. [n. d.]. Peerceptiv. https://peerceptiv.com/. Accessed: 2021-01-17.

[16] PeergradeApS. [n. d.]. Peergrade. https://www.peergrade.io. Accessed: 2019-08-18.

[17] C. Piech, J. Huang, Zhenghao Chen, C. Do, A. Ng, and D. Koller. 2013. Tuned models of peer assessment in MOOCs. *arXiv preprint arXiv:1307.2579* (2013).

[18] S. Prajapati, A. Gupta, S. K. Nigam, and S. Nath. 2020. SwaGrader: An honest effort extracting, modular peer-grading tool. In *COMAD'20*. 312–316.

[19] P. M. Sadler and E. Good. 2006. The impact of self-and peer-grading on student learning. *Educational Assessment* 11, 1 (2006), 1–31.

[20] T. Walsh. 2014. The PeerRank method for peer assessment. In *ECAI'14*. 909–914.

[21] J. Wang and N. B. Shah. 2019. Your 2 is my 1, your 3 is my 9: Handling arbitrary miscalibrations in ratings. In *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems*. International Foundation for Autonomous Agents and Multiagent Systems, 864–872.

[22] T. Wang, X. Jing, Q. Li, J. Gao, and J. Tang. 2019. Improving peer assessment accuracy by incorporating relative peer grades. *International Educational Data Mining Society* (2019).

[23] J. R. Wright, C. Thornton, and K. Leyton-Brown. 2015. Mechanical TA: Partially automated high-stakes peer grading. In *Proceedings of the 46th ACM Technical Symposium on Computer Science Education*. ACM, 96–101.

[24] H. Zarkoob, G. d'Eon, L. Podina, and K. Leyton-Brown. 2023. Better peer grading through bayesian inference. In *Proceedings of the 37th AAAI Conference on Artificial Intelligence*, Vol. 37. 6137–6144.

[25] H. Zarkoob, H. Fu, and K. Leyton-Brown. 2019. Report-Sensitive Spot-checking in Peer Grading Systems. In *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems*. International Foundation for Autonomous Agents and Multiagent Systems, 2306–2308.