Peer-grading “Explain in plain English” questions: A Bayesian calibration method for categorical answers

Binglin Chen
chen386@illinois.edu
University of Illinois
Urbana-Champaign

Matthew West
mwest@illinois.edu
University of Illinois
Urbana-Champaign

Craig Zilles
zilles@illinois.edu
University of Illinois
Urbana-Champaign

ABSTRACT

“Explain in plain English” (EipE) questions have been proposed as an important activity and assessment for studying novice programmers’ grasp of programming knowledge and their ability to communicate their understanding. However, EipE questions aren’t widely used in introductory programming courses in part because of the large grading effort required. In this paper, we present our experience of using peer grading for EipE questions in a large-enrollment introductory programming course, where students were asked to categorize other students’ responses. We developed a novel Bayesian algorithm for performing calibrated peer grading on categorical data, and we used a heuristic grade assignment method based on the Bayesian estimates. The peer-grading exercises served both as a way to coach students on what is expected from EipE questions and as a way to alleviate the grading load for the course staff. Based on four rounds of peer-grading activities, we found that students are generally capable of categorizing responses to EiPE questions and that our proposed Bayesian method is more robust than unweighted voting.

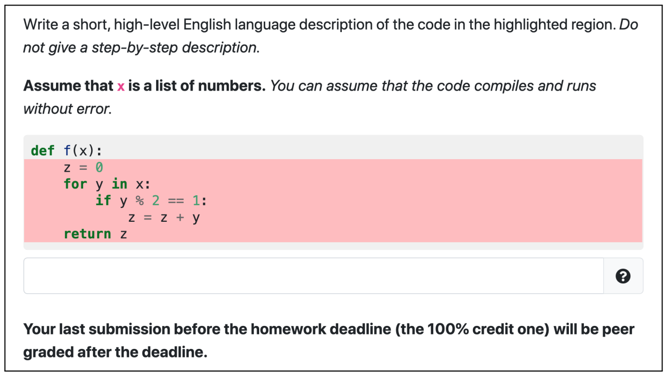
CCS CONCEPTS

• **Social and professional topics** → **Computing education**.

KEYWORDS

explain in plain English; EipE; peer grading; CS1; experience report

ACM Reference Format:
Binglin Chen, Matthew West, and Craig Zilles. 2021. Peer-grading “Explain in plain English” questions: A Bayesian calibration method for categorical answers. In *Proceedings of ACM Conference (Conference’17).* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn



Write a short, high-level English language description of the code in the highlighted region. *Do not give a step-by-step description.*

Assume that x is a list of numbers. *You can assume that the code compiles and runs without error.*

```
def f(x):
    z = 0
    for y in x:
        if y % 2 == 1:
            z = z + y
    return z
```

Your last submission before the homework deadline (the 100% credit one) will be peer graded after the deadline.

Figure 1: An example EipE question.

1 INTRODUCTION

“Explain in plain English” (EipE) questions, as exemplified in Figure 1, present students with a piece of code and ask students to describe in natural language what the code does. These questions often emphasize the desirability of holistic descriptions that capture the purpose of the code rather than line-by-line descriptions.

EipE questions have been proposed as a way to assess students’ ability to read code, a skill theorized to play an important role in learning to program [10]. Researchers have suggested that there is a hierarchy of programming skills that novices have to master: understanding the syntax (easiest), code tracing, code reading/explaining, and code writing (hardest) [12]. Students’ struggles with code writing might be due to their insufficient mastery of skills lower in the hierarchy. We discuss more on this line of work in Section 2.1.

Despite their potential utility for teaching programming, code reading tasks such as EipE questions aren’t widely used in introductory programming courses in part because of grading load concerns [7]. Unlike code tracing and code writing questions whose answers can be cleanly categorized into correct and incorrect, allowing them to be automatically graded most of the time, EipE questions involve natural language responses, which are less straightforward to grade and would typically require manual grading.

In this work, we explore peer grading as a way to both reduce the grading load for course staff and train students on the grading criteria of EipE questions in a large-enrollment university-level introductory programming course. Specifically we: 1) describe how we set up the peer-grading exercises, 2) describe a mathematically rigorous Bayesian algorithm which we believe to be the first published method of calibrated peer grading on categorical data, 3) compare students’ votes and teaching assistants’ (TAs) votes and find that their overall distributions are similar, 4) compare the performance of the proposed Bayesian method against unweighted voting and a heuristic weighted voting and find that the proposed method is more robust than unweighted voting.

Conference’17, July 2017, Washington, DC, USA
© 2021 Association for Computing Machinery.
ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

Response category	Example response
(a) a clear, correct, high-level answer	compute the sum of odd numbers in a list
(b) functionally incorrect or incomplete	The code finds the sum of all negative numbers in a list.
(c) ambiguous, not understandable, or garbage	returns the sum
(d) too low level	iterates y through the list x, adds y to z if y mod 2 equals 1, then returns z

Table 1: Student answers to EipE questions can be categorized into one of four categories: correct and having one of the three major shortcomings. For each category we show an example response to the question shown in Figure 1.

2 RELATED WORK

2.1 Hierarchy of programming skills

The idea that there is a hierarchy of programming skills was put forward by Lopez et al. in 2008 [13]. These skills range from understanding syntax (easiest), to code tracing, to code reading/explaining, to code writing (hardest) [13]. Evidence suggests that for the same piece of code, task difficulty generally increases as we move up the hierarchy (e.g., tracing bubble sort vs. reading/explaining bubble sort vs. writing a bubble sort) [12, 29], and students' performance on tasks only requiring the lower level skills is predictive of their performance on code writing tasks [5, 11, 12, 26]. For example, Lopez et al. find that students' performance on tracing and code reading questions together explain 46% of the variance in their performance on code writing questions [12]. In a later study, Lister et al. state that, while their data doesn't support the idea of a strict hierarchy, "We found that students who cannot trace code usually cannot explain code, and also that students who tend to perform reasonably well at code writing tasks have also usually acquired the ability to both trace code and explain code." [11].

Whalley et al. argue that for novices, understanding a piece of code and knowing the relevant knowledge are prerequisites to write the same piece of code [29]. Specifically, programmers need to think about code at the *relational* level rather than the multi-structural level [29]. Longitudinal studies show that students who struggle to explain code at a relational level early in the semester tend to struggle to write code later in the semester [5].

Multiple authors have argued that novice instruction should focus more on code tracing and code reading [1, 3, 4, 11, 17, 30]. Lister et al. state, "It is our view that novices only begin to improve their code writing ability via extensive practice in code writing when their tracing and explaining skills are strong enough to support a systematic approach to code writing [...] Until students have acquired minimal competence in tracing and explaining, it may be counter productive to have them write a great deal of code " [11].

Code reading tasks may be the most suitable activity for novices to learn common programming idioms. One key distinction between novice and expert programmers is that experts can process multiple syntax elements as one unit by 'chunking' [3, 8, 16, 31], which leads to less cognitive load [24]. These chunks (or *schema* in the cognitive load literature) are developed through repeated exposure to problems with common identifiable features [15], and thus can be learned more efficiently in lower cognitive load activities such as code reading as compared to code writing [25].

2.2 Peer grading

Peer grading, also referred to as peer assessment, is an assessment method where students evaluate other students' work to provide feedback, typically in the form of a numeric rating. Prior research suggests that peer grading is valid when compared to instructor grading. Specifically, Falchikov and Goldfinch conducted a meta-analysis of 48 studies published from 1959 to 1999 that focused on higher education [6]. These studies cover a wide range of subjects that include natural sciences, social sciences, engineering, medicine, and business. Falchikov and Goldfinch found that the overall correlation between student-assigned scores and instructor-assigned scores is 0.69. Similar results in other contexts have also been reported. For example, Sadler and Good reported correlations of 0.91 to 0.94 in the context of secondary education [23]; Luo et al. reported a correlation of 0.62 in a MOOC [14]; Price et al. reported correlations of 0.89 to 0.91 in a guided-inquiry conceptual physics course [20].

Various attempts have been made to make peer grading more reliable. More complex methods that account for students' bias and reliability have been proposed to replace taking the mean or median of students' ratings [9, 19]. Some researchers suggest using peer ranking because ranking responses is an easier task than assigning numerical values, which could lead to more reliable results [21, 27]. Including calibration questions is another useful technique. Students will either receive feedback on the calibration questions directly, or the information can be used to compute the final grade. This technique is adopted by widely used peer grading systems such as Calibrated Peer Review and peerScholar [18, 22].

3 METHODS

3.1 EipE grading

From past experience of grading EipE questions [2], we noticed that students' responses can generally fit into four categories, as exemplified in Table 1: (a) descriptions that correctly summarize the code at a high level unambiguously, (b) descriptions that misrepresent what the code does, either incorrect or incomplete, (c) descriptions that are ambiguous, hard to understand, or not even English, (d) descriptions that correctly describe what the code does, but are at a low level. We only consider responses that fall under (a) as correct, and would like students to recognize the three common errors.

3.2 Course context

We developed and deployed the EipE peer-grading activities in an introductory CS course for non-technical majors at a large U.S.

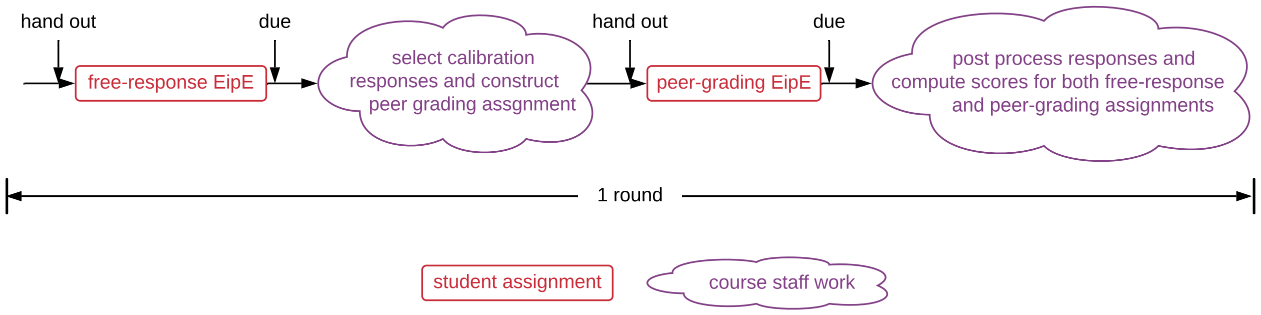

hand out
due
free-response EipE
select calibration responses and construct peer grading assgnment
hand out
due
peer-grading EipE
post process responses and compute scores for both free-response and peer-grading assignments
1 round
student assignment
course staff work

Figure 2: Workflow of 1 round of peer-grading activity.

university during Spring 2021. This large-enrollment course (capped at 750 students) introduces basic principles of programming in both Python and Excel to a population largely without any prior programming experience.

To familiarize and train students with peer-grading EipE questions, the instructor of the course asked students example peer-grading EipE questions covering all categories in Table 1 as clicker questions during lectures twice throughout the semester. The instructor elaborated on why a particular response belongs to the designated category after each of these questions.

3.3 Workflow

We implemented the peer-grading EipE activities in the same online system that students use for their homework [28]. A complete round of peer-grading EipE activity had exactly one piece of code and two phases of assignments, as illustrated in Figure 2. In the first phase, we chose a piece of code for the round and then wrote an EipE question for that piece of code, with the additional notice in the question that responses to the question were going to be peer-graded. This question was then added to the regular weekly homework assignment. An example of a question in the first phase is shown in Figure 1.

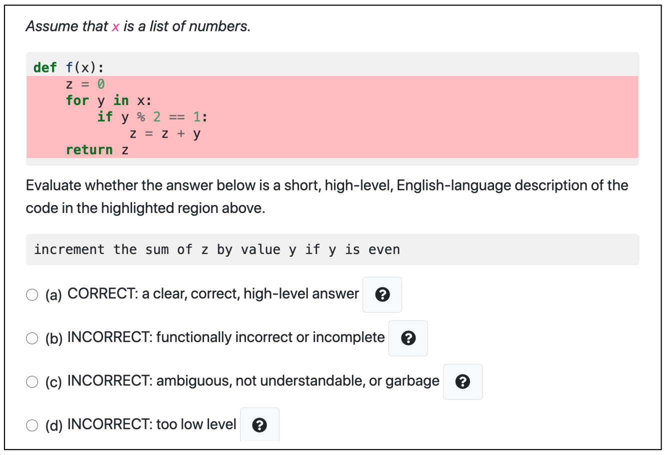
In the second phase, we constructed a peer-grading assignment consisting of only peer-grading multiple-choice questions of the format shown in Figure 3. The question presented the same piece of code and a response that students need to categorize into one of the four categories from Table 1. A typical peer-grading assignment would have a total of 16 such questions.

The process for constructing each set of these 16 questions was as follows. After the deadline of the corresponding regular homework, one TA would pull all the last submissions to the EipE question made before the homework deadline. The same TA then manually curated 10 responses for each of the four response categories, mostly by taking student responses unchanged if possible. If not enough unchanged responses were found under a category, the TA would tweak student responses to fit the target category (without replacing the original). If tweaking student responses was still not enough, the TA would write an appropriate response from scratch. These 10 curated responses per category were then checked by another TA to verify that they unambiguously belonged to the target category. Any of the curated responses that failed the check would then be replaced by the first TA until all 40 responses passed the check. We then randomly split each of the 10 responses into 2 groups of 5, and constructed 1 calibration question for each of the groups. This gave us a total of 8 calibration questions, 2 for each category, where each question would randomly draw 1 response out of the group of 5 responses for each student. The rest of the student responses that were not identical to any of the 40 curated responses were then randomly split into 8 groups, each of which was used to construct a non-calibration question, where each question would randomly draw 1 response out of the group of responses it was associated with for each student. This process results in 8 calibration questions and 8 non-calibration questions.

The calibration questions were introduced to measure each student's ability to grade according to our standard, allowing us to identify students who were either too lenient or too strict. We did not inform students of the existence of the calibration questions, and all 16 questions were displayed to each student in a different random order so that it would be hard for students to ascertain which questions were calibration questions even if they were aware of them. Unlike the EipE question in the first phase which was optional because the regular homework was designed in a way such that students don't have to answer all questions to obtain full credit, every question in the peer grading assignment was required for full credit. In total, students can earn 0.2% worth of course credit from the EipE questions in the first phase and 1.1% worth of course credit from the questions in the second phase.

4 PEER-GRADING ALGORITHM

For each round of the peer-grading activity, we wanted to automatically compute credit for both the EipE question in the regular homework and the peer-grading assignment. The algorithm that we developed is based on Bayesian inference, which yields a probability distribution over the four response categories for each of the responses to the EipE question. The main motivation for this algorithm is to take into account how students performed on calibration questions as a way to obtain a more reliable result in a mathematically rigorous (non-heuristic) way. Credit for both assignments were then automatically assigned using a heuristic based on the distribution which allows for partial credit.


Assume that x is a list of numbers.

```
def f(x):
    z = 0
    for y in x:
        if y % 2 == 1:
            z = z + y
    return z
```

Evaluate whether the answer below is a short, high-level, English-language description of the code in the highlighted region above.

```
increment the sum of z by value y if y is even
```

- (a) CORRECT: a clear, correct, high-level answer
- (b) INCORRECT: functionally incorrect or incomplete
- (c) INCORRECT: ambiguous, not understandable, or garbage
- (d) INCORRECT: too low level

Figure 3: An example peer-grading question.

We distinguish these three incorrect categories because we've found that learning to write a response that is complete, unambiguous, and at a high-level can be challenging for our students. Often they quickly master one or two of these criteria, but struggle to understand our expectations in the other dimensions. Our hope is that by evaluating other students' work using this rubric, the students will better understand our expectations and learn to construct answers that warrant full credit.

4.1 Bayesian inference

Given a response, we would like to compute a distribution over the four categories, so that it can be used heuristically later to assign credit. Let $X \in \{a, b, c, d\}$ be the random variable that represents which category the response belongs to. We have a set of students who have voted on this response in the peer-grading assignment. Denote the size of the set as n, and let $S_1, S_2, \ldots, S_n \in \{a, b, c, d\}$ be the set of random variables, with S_i being the response category chosen by the ith student. We denote the observed students votes as $s_1, s_2, \ldots, s_n$. The values that we are interested in are

$$P(X = y \mid S_1 = s_1,\ S_2 = s_2,\ \ldots,\ S_n = s_n) \tag{1}$$

for $y \in \{a, b, c, d\}$, which are the probabilities that the response belongs to category y given the student votes.

By Bayes' theorem we know that Equation 1 is equivalent to

$$\frac{P(S_1 = s_1,\ S_2 = s_2,\ \ldots,\ S_n = s_n \mid X = y)\ P(X = y)}{P(S_1 = s_1,\ S_2 = s_2,\ \ldots,\ S_n = s_n)}. \tag{2}$$

Since the denominator is the same regardless of which value y takes, we only need to consider the numerator for the purpose of computing a probability distribution among the four categories. The right term $P(X = y)$ is the prior probability of an arbitrary response belonging to category y. It can be estimated by manually categorizing a small sample (50-100) of student responses to the EipE question. The left term $P(S_1 = s_1,\ S_2 = s_2,\ \ldots,\ S_n = s_n \mid X = y)$ can be further broken down by applying the naive Bayes assumption, which states that each student's vote is independent of other students. This simplifies the left term to

$$P(S_1 = s_1 \mid X = y)\ P(S_2 = s_2 \mid X = y)\ \ldots\ P(S_n = s_n \mid X = y). \tag{3}$$

Ideally, we would like to know $P(S_i = z \mid X = y)$ for every pair of $(z, y) \in \{a, b, c, d\} \times \{a, b, c, d\}$. However, since we only had 2 calibration questions per category, obtaining reliable estimates of them is unrealistic. What we can estimate well from the calibration questions is the probability that a particular student will agree with the ground truth category, denoted as $P(S_i = X)$. When $z = y$, we simply let $P(S_i = z \mid X = y)$ take the value of $P(S_i = X)$ since the student agrees with which category the response belong to. When $z \neq y$, we split the remaining probability among the other categories proportional to the probability of the student voting for each of the remaining categories. This approach can be formulated as follows:

$$P(S_i = s_i \mid X = y) = \begin{cases} P(S_i = X) & \text{if } s_i = y, \\ (1 - P(S_i = X)) \times \frac{P(S_i = s_i)}{1 - P(S_i = y)} & \text{if } s_i \neq y, \end{cases} \tag{4}$$

where $P(S_i = z)$ is the probability that the ith student will vote for category z.

We take $P(S_i = X) = \frac{a_i + 1}{a_i + b_i + 2}$ where a_i, b_i are the number of agreements and disagreements of the ith student on calibration questions for the relevant category. This formulation is derived from Bayesian inference on a Beta distribution with the assumption that without any evidence, the student's probability of agreement follows a $\text{Beta}(1, 1)$ distribution. As for $P(S_i = z)$, we simply substitute the overall probability of students voting z for questions in the peer-grading assignment. That is, we counted the occurrences of z being voted among all students on all peer-grading questions in the round of EipE activity, and divided it by the total number of votes.

Finally, we substitute all the values and compute

$$P(S_1 = s_1 \mid X = y)\ \ldots\ P(S_n = s_n \mid X = y)\ P(X = y) \tag{5}$$

for each $y \in \{a, b, c, d\}$. This gives us a weight for each category and normalizing them to have sum 1 gives the probability distribution we want over the response categories.

4.2 Heuristic for awarding partial credit

Given the probability distribution computed with the proposed method above, we would like to be able to give partial credit for both the EipE question in the regular homework and the questions in the peer-grading assignment. Since assigning partial credit needs to take practical concerns such as overall score distribution into account, there isn't a mathematically sound way to model it. We therefore used the heuristic below.

We would like to assign a grade to a free response answers in the regular homework "proportional" to how dominant the response was categorized as correct in the peer-grading assignment. Specifically, denote the probabilities over the categories for a particular response computed by the Bayesian method as p_a, p_b, p_c, p_d. We first computed $r = \frac{p_a}{\max(p_a, p_b, p_c, p_d)}$. We gave 100% credit if r is 1, 75% credit if r is greater than 0.75, 50% credit if r is greater than 0.5, 25% credit if r is greater than 0.25, and no credit otherwise. For

Round	Round 1	Round 2	Round 3	Round 4
Fraction of students	54.7%	35.7%	43.9%	36.2%

Table 2: Fraction of students that attempted the EipE question in the first phase.

Round	Round 1				Round 2				Round 3				Round 4			
Category	a	b	c	d	a	b	c	d	a	b	c	d	a	b	c	d
TA 1	6	8	2	5	9	7	0	10	10	6	2	0	9	9	2	1
TA 2	9	7	9	7	9	9	9	8	10	4	10	6	9	7	7	5
TA 3	8	8	8	7	9	8	8	7	9	1	9	3	9	7	7	4
TA 4	9	9	1	3	9	10	0	7	10	6	1	1	9	9	2	0

Table 3: Number of TA agreements with designated category for calibration responses.

example, if the probabilities were 0.22, 0.40, 0.28, 0.10, then $r = 0.55$ and the student would receive 50% credit.

For the peer-grading assignment, we would like to award credit to a student based on how dominant the category the student picked is among all students who graded the same response. We divide the credit calculation into two equal portions. In the first portion, we award credit strictly based on the dominance of the category the student picked. For example, if the majority of students think that the response should be labeled b and the student picked c, then the student won't receive much credit for this portion. However, we would still like to award the student some credits because the student agreed with the majority that the response is incorrect, and this is what the second portion aims to achieve. Specifically, given the probabilities p_a, p_b, p_c, p_d and the ith student that has voted s_i, we computed $r_1 = \frac{p_{s_i}}{\max(p_a, p_b, p_c, p_d)}$ and $r_2 = \frac{p_a \mathbf{1}_{s_i=a} + (p_b+p_c+p_d)\, \mathbf{1}_{s_i \neq a}}{max(p_a,\, p_b+p_c+p_d)}$ where $\mathbf{1}_{s_i=a}$ is an indicator that evaluates to 1 when $s_i = a$ and 0 otherwise. We chose r_2 to be a binary version of r_1 obtained by grouping the "incorrect" categories b, c, d. Finally, we gave 50% credit if r_1 is greater than 0.75, 25% credit if greater than 0.5, 12.5% credit if greater than 0.25, and no credit otherwise. We do the same for r_2, and the sum of the two was the final credit given to the ith student. For example, if the probabilities were 0.20, 0.32, 0.40, 0.08 and the student voted on b, then $r_1 = 0.8, r_2 = 1$ and the student would receive 100% credit.

5 RESULTS

A total of four rounds of peer-grading EipE activities were administered in the later half of the semester. The fraction of students answering the EipE question in the first phase for the four rounds decreased as the semester went on as shown in Table 2. This was expected because the EipE questions were the only type of questions that didn't have instant feedback when a submission was made. Given that the regular homework was designed in such a way that students didn't have to answer all questions to obtain full credit, it was only reasonable that students chose to simply ignore the EipE question and do other questions to ensure full credit for the homework.

At the end of the semester, four TAs (including the two that were responsible for the calibration questions) independently categorized all of the students' responses and the manually curated responses to the four EipE questions. Only 10–20 random example responses from each round were discussed together by all four TAs before

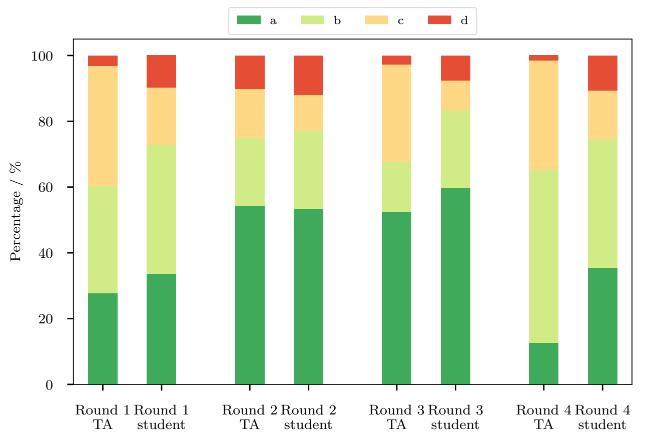


Figure 4: Raw distribution of votes among the categories by TAs versus students for each round on non-calibration responses.

each TA categorized all of the remaining responses independently. All responses for each of the rounds were randomly shuffled so that the TAs had no knowledge of which responses were for the calibration questions. In Table 3 we report the number of matches each TA had on each category of calibration responses in each round of peer-grading activities. All four TAs are CS masters/PhDs, and have been manually grading EipE questions in the course. TA 2 and TA 3 were the two TAs responsible for the curated responses of the calibration questions, and have worked on research related to EipE previously. TA 1 and TA 4 have taught the course for more than one semesters, but haven't participated in any EipE research. TA 2 and TA 3 had higher agreement with the designated category for the calibration responses. One interesting exception is category b of round 3 where both of them had lower agreement with the designated categories. In general, the TAs agreed on category a and b most of the time, and TA 1 and TA 4 had low agreements with category c and d.

To understand how the TAs' votes differ from the students' votes, we plotted the raw distribution of votes among the four categories of all non-calibration responses in Figure 4. As the figure shows, TA's overall distributions aren't very different from those of students in most cases. An interesting observation is that the TA's distribution

Metric	Jensen–Shannon				Total Variation				Top Match			
Round	1	2	3	4	1	2	3	4	1	2	3	4
Unweighted voting	0.381	0.309	0.373	0.419	0.357	0.261	0.340	0.411	0.662	0.826	0.767	0.541
Weighted voting	0.362	0.279	0.358	**0.407**	**0.338**	0.230	**0.330**	**0.403**	0.670	0.826	**0.780**	0.558
Bayesian	**0.356**	**0.215**	**0.339**	0.425	0.382	**0.228**	0.354	0.476	**0.681**	**0.843**	0.760	**0.609**

Table 4: Performance of different methods for computing probability distribution under different metrics for all four rounds. Bold numbers are the best of the columns.

varies significantly across rounds, suggesting that different EipE questions had different distributions of student responses among the categories.

We evaluated how well the proposed Bayesian method can construct a probability distribution over the four categories for each response by evaluating how close it is to the corresponding TAs' probability distribution. We included two baseline methods for comparison: *unweighted voting* and *weighted voting*. The unweighted voting method simply takes all students' votes equally to compute a probability distribution on the response. The weighted voting method is a heuristic that weighs each student's vote by a factor that is equal to the ratio of calibration questions on which the student matched the designated categories. We evaluated on three metrics: *Jensen–Shannon*, *Total Variation*, and *Top Match*, where the Jensen–Shannon distance and Total Variation distance are two commonly used distance metrics for probability distributions where lower values are better. The Top Match metric finds the categories with the highest probability in each probability distribution. If they match, then it evaluates to 1 otherwise it evaluates to 0, so a higher value is better. In the case when the four TAs are split 50%/50% on two categories, Top Match will evaluate to 1 as long as the top category in the students' distribution is one of the two categories. We applied these metrics to all responses and then aggregated the results based on the metric, method, and the round. We reported the mean for each of these combinations in Table 4, where the bold numbers represents the best value of that metric for each round. As the table shows, the unweighted voting method performed universally the worst. The Bayesian method is better under Jensen–Shannon and Top Match, whereas the weighted voting method is better under Total Variation. However, the gap between these two are not large enough to conclude that one is better or worse.

6 CONCLUSION

In this paper, we reported our experience of running peer grading on"Explain in plain English" (EipE) questions. We observed that the overall distributions of votes by students are similar to that of the TAs'. Both our proposed mathematically rigorous Bayesian method and a heuristic weighted voting outperform unweighted voting, suggesting that calibration indeed improves the peer-grading results. On our dataset the Bayesian method performed similarly with the heuristic weighted voting.

We believe that peer-graded EipE questions can both reduce the workload for the course staff and train students on what's expected in responses to EipE questions. This would enable EipE questions to be adopted with less effort, as a common reason that prevents instructors from using EipE questions is the effort required to grade them in formative assessments.

In our implementation of peer-grading EipE questions, we cast the grading problem as a multiple choice question, though in reality a response can be both functionally incorrect and low level. We believe this multiple-choice approach is sufficient for the purpose of giving formative feedback and training students to be aware of the common pitfalls, though a future direction could be to explore how to make peer grading work for "check all that apply" questions.

ACKNOWLEDGMENTS

This material is based upon work supported by the National Science Foundation under Grants No. DUE-1915257 and DUE-2121424.

REFERENCES

[1] Owen Astrachan and David Reed. 1995. AAA and CS 1: The Applied Apprenticeship Approach to CS 1. In *Proceedings of the Twenty-sixth SIGCSE Technical Symposium on Computer Science Education* (Nashville, Tennessee, USA) *(SIGCSE '95)*. ACM, New York, NY, USA, 1–5. https://doi.org/10.1145/199688.199694

[2] Binglin Chen, Sushmita Azad, Rajarshi Haldar, Matthew West, and Craig Zilles. 2020. A Validated Scoring Rubric for Explain-in-Plain-English Questions. In *Proceedings of the 51st ACM Technical Symposium on Computer Science Education*. 563–569.

[3] Michael J. Clancy and Marcia C. Linn. 1999. Patterns and Pedagogy. In *The Proceedings of the Thirtieth SIGCSE Technical Symposium on Computer Science Education* (New Orleans, Louisiana, USA) *(SIGCSE '99)*. ACM, New York, NY, USA, 37–42. https://doi.org/10.1145/299649.299673

[4] Malcolm Corney, Sue Fitzgerald, Brian Hanks, Raymond Lister, Renee McCauley, and Laurie Murphy. 2014. 'Explain in Plain English' Questions Revisited: Data Structures Problems. In *Proceedings of the 45th ACM Technical Symposium on Computer Science Education* (Atlanta, Georgia, USA) *(SIGCSE '14)*. ACM, New York, NY, USA, 591–596. https://doi.org/10.1145/2538862.2538911

[5] Malcolm Corney, Raymond Lister, and Donna Teague. 2011. Early Relational Reasoning and the Novice Programmer: Swapping As the "Hello World" of Relational Reasoning. In *Proceedings of the Thirteenth Australasian Computing Education Conference - Volume 114* (Perth, Australia) *(ACE '11)*. Australian Computer Society, Inc., Darlinghurst, Australia, Australia, 95–104. http://dl.acm.org/citation.cfm?id=2459936.2459948

[6] Nancy Falchikov and Judy Goldfinch. 2000. Student peer assessment in higher education: A meta-analysis comparing peer and teacher marks. *Review of educational research* 70, 3 (2000), 287–322.

[7] Max Fowler, Binglin Chen, and Craig Zilles. 2021. How should we 'Explain in plain English'? Voices from the Community. In *ACM Conference on International Computing Education Research*. 69–80.

[8] Fernand Gobet, Peter CR Lane, Steve Croker, Peter CH Cheng, Gary Jones, Iain Oliver, and Julian M Pine. 2001. Chunking mechanisms in human learning. *Trends in cognitive sciences* 5, 6 (2001), 236–243.

[9] John Hamer, Kenneth TK Ma, and Hugh HF Kwong. 2005. A method of automatic grade calibration in peer assessment. In *Proceedings of the 7th Australasian conference on Computing education-Volume 42*. 67–72.

[10] Raymond Lister, Elizabeth S Adams, Sue Fitzgerald, William Fone, John Hamer, Morten Lindholm, Robert McCartney, Jan Erik Moström, Kate Sanders, Otto Seppälä, Beth Simon, and Lynda Thomas. 2004. A multi-national study of reading and tracing skills in novice programmers. *ACM SIGCSE Bulletin* 36, 4 (2004), 119–150.

[11] Raymond Lister, Colin Fidge, and Donna Teague. 2009. Further Evidence of a Relationship Between Explaining, Tracing and Writing Skills in Introductory Programming. In *Proceedings of the 14th Annual ACM SIGCSE Conference on Innovation and Technology in Computer Science Education* (Paris, France) *(ITiCSE '09)*. ACM, New York, NY, USA, 161–165. https://doi.org/10.1145/1562877.1562930

[12] Mike Lopez, Jacqueline Whalley, Phil Robbins, and Raymond Lister. 2008. Relationships between reading, tracing and writing skills in introductory programming. In *Proceedings of the Fourth International Workshop on Computing Education Research*. ACM, 101–112.

[13] Mike Lopez, Jacqueline Whalley, Phil Robbins, and Raymond Lister. 2008. Relationships between reading, tracing and writing skills in introductory programming. In *International Workshop on Computing Education Research*. ACM, 101–112.

[14] Heng Luo, Anthony Robinson, and Jae-Young Park. 2014. Peer grading in a MOOC: Reliability, validity, and perceived effects. *Online Learning Journal* 18, 2 (2014).

[15] Sandra P Marshall. 1995. *Schemas in problem solving*. Cambridge University Press.

[16] Katherine B McKeithen, Judith Spencer Reitman, Henry H Rueter, and Stephen C Hirtle. 1981. Knowledge organization and skill differences in computer programmers. *Cognitive Psychology* 13, 3 (1981), 307–325.

[17] Laurie Murphy, Renée McCauley, and Sue Fitzgerald. 2012. 'Explain in Plain English' Questions: Implications for Teaching. In *Proceedings of the 43rd ACM Technical Symposium on Computer Science Education* (Raleigh, North Carolina, USA) *(SIGCSE '12)*. ACM, New York, NY, USA, 385–390. https://doi.org/10.1145/2157136.2157249

[18] Dwayne E Paré and Steve 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.

[19] Chris Piech, Jonathan Huang, Zhenghao Chen, Chuong B. Do, Andrew Y. Ng, and Daphne Koller. 2013. Tuned Models of Peer Assessment in MOOCs.. In *Proceedings of the 6th International Conference on Educational Data Mining*. 153–160.

[20] Edward Price, Fred Goldberg, Steve Robinson, and Michael McKean. 2016. Validity of peer grading using Calibrated Peer Review in a guided-inquiry, conceptual physics course. *Physical Review Physics Education Research* 12, 2 (2016), 020145.

[21] Karthik Raman and Thorsten Joachims. 2014. Methods for ordinal peer grading. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. 1037–1046.

[22] Ralph Robinson. 2001. Calibrated Peer Review™ an application to increase student reading & writing skills. *The American Biology Teacher* 63, 7 (2001), 474–480.

[23] Philip M Sadler and Eddie Good. 2006. The impact of self-and peer-grading on student learning. *Educational assessment* 11, 1 (2006), 1–31.

[24] John Sweller. 2011. Cognitive Load Theory. In *Psychology of learning and motivation*. Vol. 55. Elsevier, 37–76.

[25] John Sweller, Jeroen JG Van Merrienboer, and Fred GWC Paas. 1998. Cognitive architecture and instructional design. *Educational psychology review* 10, 3 (1998), 251–296.

[26] Anne Venables, Grace Tan, and Raymond Lister. 2009. A Closer Look at Tracing, Explaining and Code Writing Skills in the Novice Programmer. In *Proceedings of the Fifth International workshop on Computing Education Research*. ACM, 117–128.

[27] Andrew E Waters, David Tinapple, and Richard G Baraniuk. 2015. BayesRank: A Bayesian approach to ranked peer grading. In *Proceedings of the 2nd ACM Conference on Learning @ Scale*. 177–183.

[28] Matthew West, Geoffrey L. Herman, and Craig Zilles. 2015. PrairieLearn: Mastery-based Online Problem Solving with Adaptive Scoring and Recommendations Driven by Machine Learning. In *2015 ASEE Annual Conference & Exposition*. ASEE Conferences, Seattle, Washington.

[29] Jacqueline Whalley, Raymond Lister, Errol Thompson, Tony Clear, Phil Robbins, P K Ajith Kumar, and Christine Prasad. 2006. An Australasian study of Reading and Comprehension Skills in Novice Programmers, using the Bloom and SOLO Taxonomies. *Eighth Australasian Computing Education Conference (ACE2006)* (01 2006).

[30] Susan Wiedenbeck. 1985. Novice/expert differences in programming skills. *International Journal of Man-Machine Studies* 23, 4 (1985), 383 – 390. https://doi.org/10.1016/S0020-7373(85)80041-9

[31] Leon E. Winslow. 1996. Programming Pedagogy&Mdash;a Psychological Overview. *SIGCSE Bull.* 28, 3 (Sept. 1996), 17–22. https://doi.org/10.1145/234867.234872