Using a Glicko-based Algorithm to Measure In-Course Learning

Rachel Reddick
Coursera, Inc.
381 E Evelyn Ave
Mountain View, California 94041
rreddick@coursera.org

ABSTRACT

One significant challenge in the field of measuring ability is measuring the current ability of a learner while they are learning. Many forms of inference become computationally complex in the presence of time-dependent learner ability, and are not feasible to implement in an online context. In this paper, we demonstrate an approach which can estimate learner skill over time even in the presence of large data sets. We use a rating system derived from the Elo rating system and its relatives, which are commonly used in chess and sports tournaments. A learner's submission of a course assignment is interpreted as a single match. We apply this approach to Coursera's online learning platform, which includes millions of learners who have submitted assignments tens of millions of times in over 3000 courses. We demonstrate that this provides reliable estimates of item difficulty and learner ability. Finally, we address how this scoring framework may be used as a basis for various applications that account for a learner's ability, such as adaptive diagnostic tests and personalized recommendations.

Keywords

Adaptive learning, student modeling, knowledge tracing, student ability estimation

1. INTRODUCTION

A requirement of any adaptive learning system is the simultaneous understanding of learner skill and item difficulty. Such measurements also enable ordering content by measured difficulty and recommending content and assessments appropriate for a learner's degree of skill, among other applications.

Item Response Theory, one common method for obtaining skill estimates in a testing context, requires assuming a fixed skill for the learner. While valid during a single exam, this assumption fails for learners who are learning during a course. Some techniques can rigorously handle skills that change over time, such as knowledge tracing [2] and performance factor analysis [6], but they are computationally intensive. For Coursera's dataset, applying these techniques would require computing results for millions of learners across thousands of courses. Further, because learners in an online platform can benefit from visibility into their own skills, online updates are desirable. Under these conditions, many approaches become computationally intractable.

To address this problem, we turned to the Elo and related rating systems, which are commonly used in chess tournaments and for team ratings in many sports [7, 9]. Following the example of [7], we treat learners and course items as players in a tournament. Each attempt by a learner at passing an item is a match.

In this paper, we demonstrate the application of the Glicko rating system [3], a variant of the Elo rating system which incorporates uncertainty, to Coursera's unique dataset. We maintain a focus on the most well-measured skills on our platform. We show that we can obtain reasonable and reliable estimates of learner skill and item difficulty. Finally, we discuss planned applications and next steps.

2. BACKGROUND

Coursera is an online learning platform, which offers courses in partnership with the universities and companies that create them. In addition to single courses, Coursera also provides "specializations," which are groupings of 3-10 courses that are usually (but not always) intended to be taken in sequence. Coursera has over 3300 different courses available, covering a broad range of subjects.

Instructors who create courses on Coursera can provide a rough estimate of the difficulty ("beginner", "intermediate", "advanced"), but this is generally not enough to establish prerequisites or help a learner know if they are ready to start a course. Since the label is at the course level, skill-related nuance is lost. For example, a course may teach both intermediate statistics and introductory programming. These labels are also insufficient for a learner to determine whether it would be more valuable to them to jump in to a course or specialization halfway through, rather than start at the beginning. Therefore, it's useful to estimate content difficulty independent of instructor labels. Personalizing this support to individual learners requires estimating their degree of skill as well.

Rachel Reddick "Using a Glicko-based Algorithm to Measure In-Course Learning" In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019),* Collin F. Lynch, Agathe Merceron, Michel Desmarais, & Roger Nkambou (eds.) 2019, pp. 754 - 759

Many other people have previously estimated content difficulty and learner skill. The most fundamental include approaches based on Item Response Theory (IRT; e.g., [1]). However, these are not suitable in Coursera's context because they generally assume constant learner skill, as in a testing environment. Although alternatives such as Bayesian knowledge tracing [5] have potential, the methods are difficult to scale up from a few tens of thousands of item attempts to over ten million for a single skill in Coursera's case. Further, the authors of [5] also note the confounding effect of learners with stronger skills working on more challenging problems, which systematically underestimates the difficulty of advanced content, even in some knowledge tracing approaches.

Therefore, we follow the work of [7] and consider the Elo rating system. The Elo system is commonly used in chess tournaments and other competitions. The general principle is that every player has a score, which is updated after matches. Updates will be large if the outcome of a match is unexpected. If a novice player defeats a master, the novice's score will have a large increase and the master will have a large decrease. On the other hand, if a master defeats a novice, the updates for both will be small or negligible. This method may be applied to learners and items in courses (such as exams), where each learner and item is interpreted as a player. A new learner who "defeats" a hard item in the Machine Learning skill will gain a large increase to that skill, but not lose much from their score if they fail. Because that ability differs from other skills (such as Management), a learner should have different scores for each possible skill. Items also need to be associated with the relevant skills.

In contrast to [7], we focus on the Glicko variant of the Elo rating system [3]. This variant uses an approximated Bayesian framework, which incorporates uncertainty in the rating. This is particularly helpful for understanding learners who have not yet completed very many items. It also supports estimation of ability across populations by enabling weighted averages based on the uncertainty. The Glicko scoring system has the drawback that it can't be easily adapted to use more complex IRT solutions, such as those for multiple choice questions with a non-zero success probability even in the case of very low skill. However, because we focus on full assignments rather than individual questions, this will not be a serious limitation.

3. DATA

Coursera has over 38 million registered learners on its platform. We draw our data from late 2014 onward.

We also have a framework for automatically tagging skills to courses. We will focus on those subjects – business, computer science, and data science – where the skills framework is best calibrated. In this two sections, we briefly describe how we tag skills to courses, and how we process the course item data that we use.

3.1 Associating Content with Skills

Based on Wikipedia's hierarchy of topics and our own human curation, we developed a hierarchy of more than 40,000 skills, over 13,000 of which are tagged to courses in the Coursera catalog (see [8] for additional details). The most popular of these skills are listed in Table 1, out of a total of 26 available. These broad skills all have a set of subskills within the skill hierarchy.

Individual skills are tagged to courses based on a combination of crowd-sourcing and machine learning. Learners who complete a course are asked to tell us what skills they learned. This information is used as the target variable of a machine-learning model, which estimates the likelihood of tagging based on course content features. The actual tag rate and machine learning prediction are combined to create a single relevance score, giving results for both popular courses and unpopular courses with few crowd-sourced tags.

For our broad skills of interest, we treat a skill as tagged to a course if any of its subskills in the course's subject area (e.g., data science) are tagged to the course.

Using this approach, for this set of skills, we obtain a total of about 1400 courses tagged with skills. Of these, about 1000 are tagged with more than one skill.

3.2 Item Attempts

We define an item attempt as a learner submission. On the Coursera platform, items include exams with multiple-choice or text answers, programming assignments that require submitting code, and peer review assignments that are graded by other learners. Most, but not all, are graded. In this analysis, we include only items that are graded. Learners are typically able to retake items up to a maximum number of attempts per day.

Learners may retake an item for many reasons, from trying again to technical issues. These repeated attempts are often uninformative. For this reason, we reduce learner attempts to those that are either the first attempt or a later attempt that does not have the same pass/fail outcome. Thus, most learner-item interactions are of the form pass (with no further attempts), fail (with no further attempts), or fail followed by pass at a later time.

This approach has the secondary benefit that if the skill scores are surfaced to learners in the future, there is no longer a motive to repeatedly submit a passed item in order to game the system to get a higher score.

We also remove all attempts at items that all learners pass on the first attempt, since these items are not informative.

We currently do not estimate scores for individual questions or sub-parts within items. Using entire items has the advantage of being able to easily include atypical items, such as programming assignments. Expanding to obtain scores for individual exam questions is left for future work.

4. IMPLEMENTATION

Our implementation of the Glicko scoring system generally followed [3], with several modifications.

Most importantly, the Glicko system assumes that players encounter each other during a tournament, during which individual scores can be assumed to be roughly constant. This is the "rating period" over which matches are accu-

Table 1: Popular Skills

Skill	Number of Attempts	Number of Courses
Statistical Programming	17,840,101	157
Machine Learning	134,52,639	60
Computer Programming	12,940,557	383
Software Engineering	9,287,216	245
Artificial Intelligence	7,900,817	127
Management	6,104,244	386

mulated, and after which an update to the scores is made. However, scores may change rapidly for learners who are learning within a course. Learners watch course lectures or review supplementary material about 40% of the time between subsequent submissions. Therefore, we are forced to use a rating period that is only one "match" long – one item attempt. In this case, the update equations from [3] reduce to, for a single match:

$$\mu' = \mu + \frac{1}{1/\sigma^2 + 1/\delta^2} g(\sigma_o^2)\{s - E(s|\mu, \mu_o, \sigma_o^2)\} \quad (1)$$

$$\sigma'^2 = \left(\frac{1}{\sigma^2} + \frac{1}{\delta^2}\right)^{-1} \quad (2)$$

Where:

$$g(\sigma^2) = \frac{1}{\sqrt{1 + 3\sigma^2/\pi^2}}$$

$$E(s|\mu, \mu_o, \sigma_o^2) = \frac{1}{1 + \exp\left[-g(\sigma_o)(\mu - \mu_o)\right]}$$

$$\delta^2 = \left[g(\sigma_o^2)^2 E(s|\mu, \mu_o, \sigma_o^2)\{1 - E(s|\mu, \mu_o, \sigma_o^2)\}\right]^{-1}$$

In these expressions, μ is the initial score, μ' is the updated score, σ is the initial uncertainty in terms of standard deviation, and σ' is the updated uncertainty. The values with a subscript 'o' are those for the opponent. If the "player" is a learner, then the opponent is an item (and vice versa). s is the outcome of the match. For a learner, passing the item is a 'win', and $s = 1$, and failing is a loss with $s = 0$. The reverse is true for items.

Our difficulty scores for items are based entirely on the scoring method described above. Instructor provided difficulty labels were not used as a feature in this framework.

Importantly, the learner's score carries over from one course to another, so long as both courses teach the same skill. For example, when a learner finishes a course and starts another, their score for the skill at the end of that course is used as their starting point for that skill in the new course. If the courses teach different skills, then the learner's score for the new skill taught in the second course will start at zero, unaffected by their score in previously learned skills.

Finally, to obtain high-quality scores for all learners, we find that we need to run the Glicko scorer twice. In the first run, we set both learners and items to have prior scores of 0

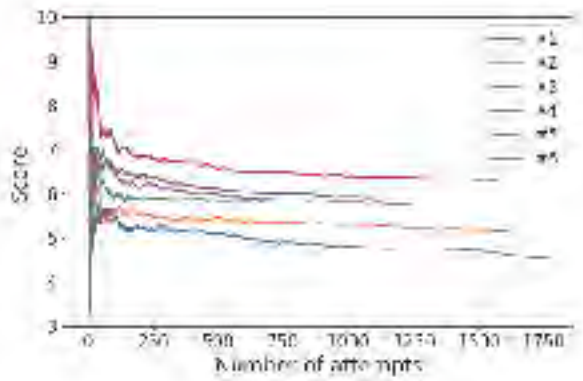

Figure 1: Scores for course items generally converge after many attempts. This example shows several items from the Practical Reinforcement Learning course, in the Machine Learning skill. The item scores rapidly converge to stable values, with a small amount of negative drift. For clarity, we only show the first few items in the course.

and uncertainties of 5, to allow large updates initially and to prevent questions with well-established scores to avoid experiencing large perturbations from new learners. This provides good estimates of item difficulty, but means that learners who took the same item early in the data, rather than late, are matched up against items whose scores may be far from reasonable values, especially for less popular content. It also means that a learner who has completed the same course, but early in our data, may not have the same score as a learner who performed in exactly the same way in the same course near the end.

Thus, all learner scores are reported based on the Glicko scorer a second time, using the item scores and uncertainties from the first run. These values for items are held fixed. The items scores for each skill are offset so that the fifth percentile item has a score of zero. Learners start with prior scores of zero and prior uncertainty of 1.0, which avoids excessive initial swings in score.

In principle, it is possible to construct a more accurate learner prior score based on information they have given us (e.g., education history) and what course they are starting. However, this could also lead to gaming of the system (by providing false information or trying a hard course first to obtain a better score), may be biased against learners with nontraditional backgrounds, and introduces additional complexity.

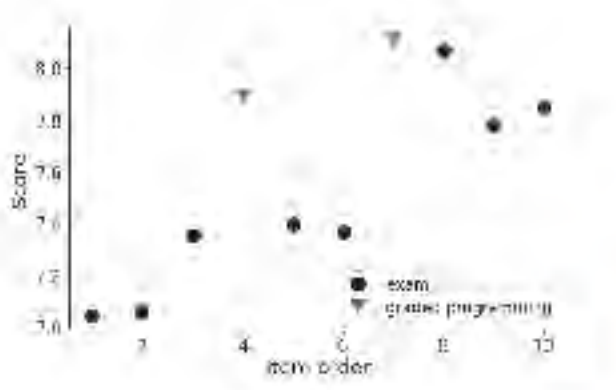

Figure 2: Comparison of final item scores within a single course (in the Machine Learning skill). As expected, later items are generally more difficult than earlier ones. The method also successfully captures differences in difficulty among different types of items – graded programming assignments tend to be more challenging than exams.

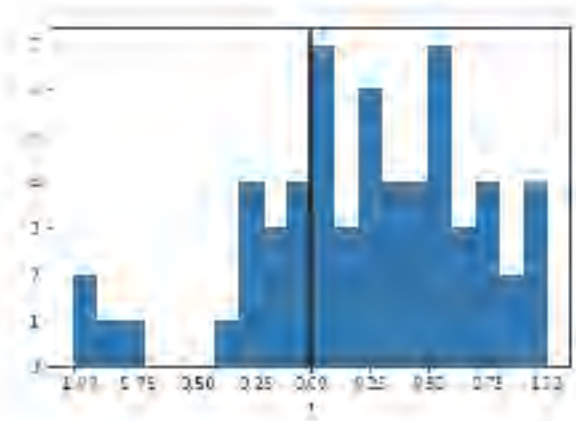

Figure 3: Histogram showing the distribution of the correlation between item difficulty and item order for each course in the Machine Learning skill. Extreme correlation (or anti-correlation) generally comes from courses with few items.

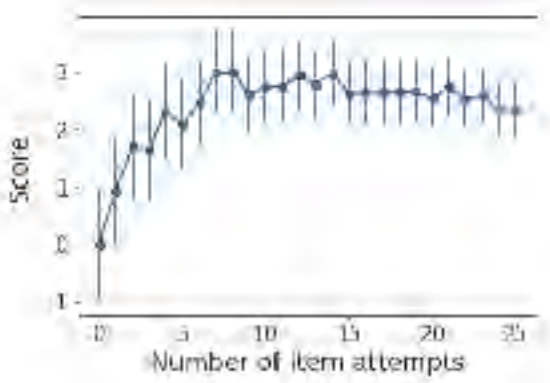

Figure 4: learner score while progressing through a course. This plot shows a learner progressing through a Machine Learning course, and completing it. Note the occasional small drop in score when the learner initially fails an item. Error bars are $\pm 1\sigma$.

5. RESULTS

We show the convergence results for a few items in a typical course in Fig. 1. This is the Practical Reinforcement Learning course, within the Machine Learning skill. As we might expect, items later in the course are typically more challenging, but not always. This reflects variation in difficulty of sub-topics within the course.

Although the item scores clearly converge within the first few hundred attempts, there is also a long-term negative drift in the scores. This is a known occurrence within Elo frameworks (for examples in chess, see [4, 10]). In the case of chess, players generally begin playing as novices and stop playing as masters, with a higher skill than they started. The pressure of new players always starting with low scores leads to ratings deflation over time, and chess tournament rating systems often include corrective factors for this reason. In our case, a similar effect occurs because learners generally start with lower scores than those of the items.

The degree of drift over a few hundred attempts is typically of the same order as the estimated uncertainty (around 0.2 for the first item in the example course). In the future, we may add a correction for this drift effect, but it does not currently have a strong effect on our results.

For clarity, we note that convergence is only expected or desirable for item scores. Items within courses are generally not modified over time, and therefore, the difficulty of the item should not change. In contrast, learner scores can be expected to change rapidly as they progress through course content.

When we examine the final item scores more closely, we find that there are some differences by item type. Programming assignments are often more difficult than regular exams (see Fig. 2), in agreement with our expectations for these items.

Even with this variation and within-topic variability, we find that difficulty does generally increase from the beginning to end of a course. Across all skills, the mean correlation between item order and item difficulty within a course is 0.25. The distribution of correlations for a single skill, Machine Learning, is shown in Fig. 3.

We also find that scores across different courses make sense. For example, based on median item score within the Machine Learning skill, the "Google Cloud Platform Big Data and Machine Learning Fundamentals" course is the easiest. This is the first course in a sequence of courses which are intended to be introductory. Conversely, the hardest course is "Probabilistic Graphical Models 3: Learning", the final course in a series of advanced Machine Learning courses.

A representative example of a learner gaining the Machine Learning skill is shown in Fig. 4. The learner rapidly increases in score to match the initial difficulty of the course, followed by more incremental increases later on. This is

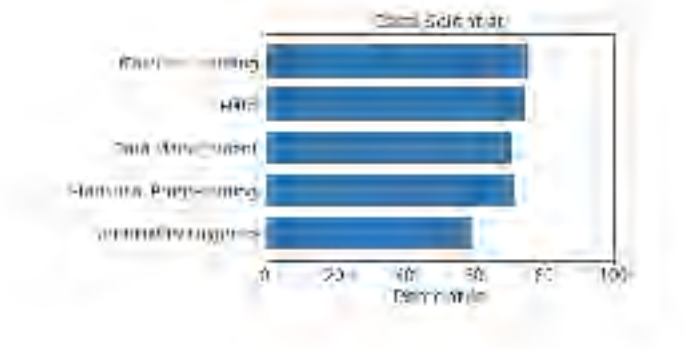

Figure 5: A profile showing the top 5 most relevant skills for a data scientist and the typical degree of proficiency. In this example, the skill score is shown as a percentile compared to the rest of the learners on Coursera.

fairly typical of learners moving through a single course.

6. FUTURE APPLICATIONS

These difficulty and ability scores, in combination with our skills tagging framework, unlocks many applications:

1. **Learner skill profiles**. These surface a summary of each learner's measured abilities. Providing this information to learners would enable them to better understand their own skills and how much they have learned so far. These profiles would update online to immediately reflect recently submitted items.
2. **Career skill profiles**. In contrast to a learner skill profile, a career skill profile shows the skills important to a career and the necessary degree of proficiency. These profiles can be built based on the skills of learners on Coursera who are already in these careers. Then these profiles can be compared against a learner profile, to allow the learner to see what they still need to enter their desired career.
3. **Recommendations by difficulty**. Given knowledge of a learner's current and desired ability in a skill, we can recommend courses that teach that skill which are at the right level for that individual.
4. **Adaptive diagnostics**. With difficulty scores for course items, it is possible to extract questions from course content and use them as the question bank for an adaptive diagnostic, which provides questions close to a test taker's estimated skill. If learners take diagnostics as a pre-test, before taking any courses, this can support the recommendations by difficulty discussed above.
5. **Review recommendations**. If a learner struggles in part of a course, a recommendation to review could find lectures or reading, potentially in another course entirely, which teaches the same skill at an easier difficulty.

We show an early example of a career skill profile for a Data Scientist in Fig. 5. These initial results align with our expectations, in that more fundamental skills for the career (e.g., Machine Learning) are required at a higher relative skill than the more niche skill of Artificial Intelligence.

We would expect this approach to generalize well to other data sets, if the fundamental data can be constructed as attempts by learners at passing items.

For simplicity, our current approach omits the addition of uncertainty for time passed since the last scoring update. In the original Glicko model, this incorporates how a player's skills may have increased due to practice or decayed due to lack of use over time. In the future, it may be valuable to restore this factor, with additional uncertainty for learners who have spent a significant time perusing course material, or who have spent a long period away from course content.

Similarly, this approach ignores cases where multiple skills are related or relevant. For example, Machine Learning and Artificial Intelligence are closely related. However, because we treat each skill in isolation, a learner who has only taken courses in Machine Learning will not have an estimated score for Artificial Intelligence. Similarly, an item in a course may require more than one skill to pass the item. In future work, we will consider multiple-skill approaches that can address these cases.

7. CONCLUSIONS

In sum, we find that using the Glicko rating system in an educational context produces reasonable scores for learners and items. The approach successfully mitigates the potential bias from more skilled learners encountering more difficult items.

We plan to use these results in many applications in the future to improve the learning experience for learners on Coursera.

8. ACKNOWLEDGMENTS

I would like to thank my colleagues, Vinod Bakthavachalam and Alan Hickey for providing valuable input on the methods and feedback on the contents of this paper. I would also like to thank Emily Sands for her feedback and support.

9. REFERENCES

[1] H. T. Binh and B. T. Duy. Student ability estimation based on irt. *2016 3rd National Foundation for Science and Technology Development Conference on Information and Computer Science*, pages 56–61, 2016.

[2] A. Corbett and J. Anderson. Knowledge tracing: Modeling the acquisition of procedural knowledge. *User modeling and user-adapted interaction*, 4:253–278, 1994.

[3] M. E. Glickman. Parameter estimation in large dynamic paired comparison experiments. *Appl. Statist.*, 48:377–394, 1999.

[4] H. Goldowsky. A conversation with mark glickman. *Chess Life*, Oct 2006.

[5] J. González-Brenes, Y. Huang, and P. Brusilovsky. General features in knowledge tracing: Applications to multiple subskills, temporal item response theory, and expert knowledge. *Proceedings of the 7th International*

Conference on Educational Data Mining, pages 84–91, 2014.

[6] P. Pavlik, H. Cen, and K. Koedinger. Performance factors analysis-a new alternative to knowledge tracing. In *Proc. of Artificial Intelligence in Education*. IOS Press, 2009.

[7] R. Pelánek. Applications of the elo rating system in adaptive educational systems. *Computers and Education*, 2016.

[8] E. G. Sands. How our skills graph is helping learners find the right content to reach their goals, July 2018.

[9] N. Silver. Our nba player projections are ready for 2018-19. *FiveThirtyEight*, June 2018.

[10] World Chess Federation. *FIDE Rating Regulations*.