A System for Scalable and Reliable Technical-Skill Testing in Online Labor Markets

Maria Christoforaki, Panagiotis G. Ipeirotis

New York University
New York, NY

Abstract

The emergence of online labor platforms, online crowdsourcing sites, and even Massive Open Online Courses (MOOCs), has created an increasing need for reliably evaluating the skills of the participating users (e.g., "does a candidate know Java") in a scalable way. Many platforms already allow job candidates to take online tests to asses their competence in a variety of technical topics. However the existing approaches face many problems. First, cheating is very common in online testing without supervision, as the test questions often "leak" and become easily available online along with the answers. Second, technical-skills, such as programming, require the tests to be frequently updated in order to reflect the current state-of-the-art. Third, there is very limited evaluation of the tests themselves, and how effectively they measure the skill that the users are tested for.

In this article we present a platform, that continuously generates test questions and evaluates their quality as predictors of the user skill level. Our platform leverages content that is already available on question answering sites such as Stack Overflow and re-purposes these questions to generate tests. This approach has some major benefits: we continuously generate new questions, decreasing the impact of cheating, and we also create questions that are closer to the real problems that the skill holder is expected to solve in real life. Our platform leverages the use of Item Response Theory to evaluate the quality of the questions. We also use external signals about the quality of the workers to examine the external validity of the generated test questins: Questions that have external validity also have a strong predictive ability for identifying early the workers that have the potential to succeed in the online job marketplaces. Our experimental evaluation shows that our system generates questions of comparable or higher quality compared to existing tests, with a cost of approximately $3 to $5 dollars per question, which is lower than the cost of licensing questions from existing test banks, and an order of magnitude lower than the cost of producing such questions from scratch using experts.

Keywords: Test Question Generation, Test Question Evaluation, Technical-Skill

Email addresses: mc3563@nyu.edu (Maria Christoforaki), panos@stern.nyu.edu (Panagiotis G. Ipeirotis)

1. Introduction

Today, increasingly more skilled labor activities are carried out online. Online labor markets, such as eLance-oDesk and Freelancer, are platforms that connect workers with relevant employers.[1] These computer-mediated marketplaces can eliminate geographical restrictions, help participants find desirable jobs, guide workers through complex goals, and better understand workers' abilities. Broadly, online labor markets offer participants the opportunity to chart their own careers, pursue work that they find valuable, and do all this at a scale that few companies can today. Spurred by this revolution, some predict that remote work will be the norm rather than the exception within the next decade [3]. One major challenge in this setting is to build skill assessment systems that can evaluate and certify the skills of workers reliably, in order to facilitate the job matching process. Online labor markets currently rely on two forms of assessment mechanisms: *reputation systems* and *skill certification*.

Reputation systems are widely used for instilling trust among the participants [4, 5]. A reputation system for an online labor market computes a reputation score for each worker based on a collection of ratings by employers that have hired them in the past. However, existing reputation systems are better-suited for markets where participants engage in a large number of transactions (e.g., selling electronics, where a merchant may sell tens or hundred of items in a short period of time). Online labor inherently suffers from data sparseness: most work engagements require at least a few hours of work, and many last for weeks or months. As a result, there are many participants that have only minimal number of feedback ratings, which is a very weak reputation signal.[2] Unfortunately, the lack of reputation signals creates a cold-start problem [9]: workers cannot get jobs because they do not have feedback, and therefore cannot get feedback that would help them to get a job. In a worst case scenario, such markets may become "markets for lemons," [10] forcing the departure of high-quality participants, leaving only low-quality workers as potential entrants.

An alternative approach to instill trust is to use skill certifications. In offline labor markets, educational credentials are often used to signal the quality of the participants and avoid the cold-start problem [11]. In global online markets, credentialing is much trickier: verifying educational background is difficult, and knowledge of the quality of the educational institutions on a global scale is limited. Given the shortcomings of using educational credentials in a global setting, many online labor markets resort to using *skill testing* as means of assessment. So today most online labor markets offer their own certification mechanisms. The goal of these tests is to certify that a given worker indeed possesses a particular skill. For example, eLance-oDesk and vWorker

[1] Online labor markets require more high level skills than microtask crowdsourcing markets [1, 2].

[2] Crowdsourcing research has recently examined the use of peer assessment as an additional form of reputation, focusing on techniques for getting crowd members to evaluate each other [6, 7]. The hope is that peer assessment can lead to better learning outcomes as well [8]. Unfortunately, these systems still have large variance in final assessment scores, which makes them a poor match for certification and qualification.

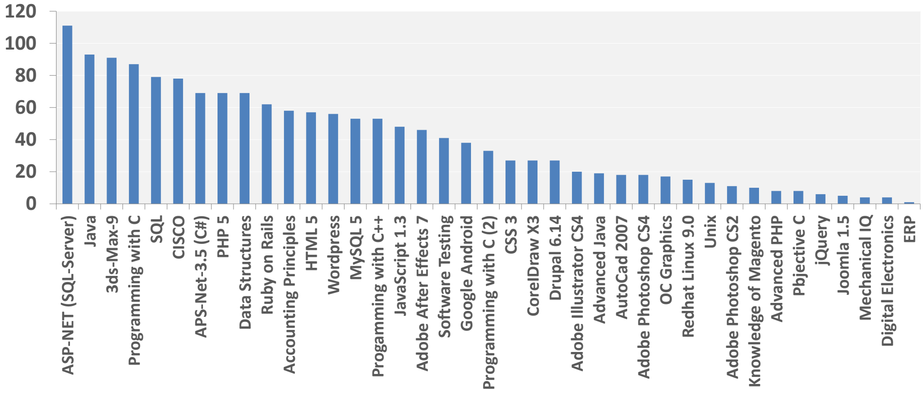


Figure 1: Number of URLs containing solutions to tests offered by eLance-oDesk and Freelancer (the two biggest online labor marketplaces). Each bar of the X-axis represents a test and the Y-axis denotes the number of identified URLs that contain the test-questions along with their answers.

allow workers to take online tests that assess the competency of these contractors across various skills (e.g., Java, Photoshop, Accounting, etc.) and then allow the contractors to display the achieved scores and ranking in their profile. Similarly, crowdsourcing companies, such as CrowdFlower and Mechanical Turk, are certifying the ability of contractors to perform certain tasks (e.g., photo moderation, content writing, translation) and allow employers to restrict recruiting to the population of certified workers. Unfortunately, online certification of skills is still problematic for a number of reasons with cheating being one of the biggest challenges.

The tests currently used by online labor platforms are usually licenced from companies such as ExpertRating[3] that pay domain experts to write test-questions; hence tests are frequently used and are accessible online. That often allows test takers to "leak" the tests and their answers become widely available on the web. Figure 1 illustrates a number of websites that contain solutions for the some of the popular tests[4] available on eLance-oDesk and Freelancer. For example, leaked questions for the ASP-NET test were identified by our crawlers in more than a hundred websites, and, correspondingly, we found more than ninety websites with leaked questions for the Java test. Needless to say, the reliability of the tests for which answers are easily available through a web search is questionable.

Furthermore, it is common, even for expert organizations, to create questions with errors or ambiguities, especially if the test questions have not been properly assessed and calibrated with relatively large samples of test takers [12]. Such problematic questions introduce noise into the user-evaluation process, hindering the correct assessment of the users's skill, and therefore need to be identified and excluded from the user-

[3] `http://www.expertrating.com/`

[4] Sites such as `http://1faq.com/` and `http://www.livejar.info/`, are a couple of examples of the offenders.

evaluation process.

Finally, many people question the value of the existing tests [13, 14, 15, 16, 17] as long-term predictors of performance, indicating that questions are calibrated only for *internal validity* (how predictive a question is about the final test score) and not for *external validity* (how predictive the question is for the long-term performance of the test taker). For example, static question banks, which are currently licenced by online labor platforms, contain questions related to fast evolving topics such as programming frameworks, which quickly become outdated. This question is particularly acute for online labor markets, as there is little research that examines whether testing and certifications are predictive of success in the labor market.

In this paper, we describe our system, which leverages content generated on popular question and answer (Q/A) sites, such as Stack Overflow, and uses these questions and answers as a basis for creating test questions. In particular, our system mines questions from Q/A sites like Stack Overflow and selects questions that could serve as good *test questions* for a particular skill. Our system is algorithmically identifying threads that are promising for generating high-quality assessment questions, and then uses a crowdsourcing system to edit these threads and transform them into multiple-choice test questions. To assess the quality of the generated questions, we employ Item Response Theory [18] and examine not only how predictive each question is regarding the internal consistency to the test [19, 20], but also examine the correlation with future real-world market-performance metrics, such as hiring rates, achieved wages, and so on, using the oDesk marketplace as our experimental testbed for evaluation.

Essentially, our system is composed of two main parts that can also function independently: the *question generation* and the *question evaluation* component. We introduce the following novel aspects for question generation:

- By utilizing Q/A threads as question seeds, we can continuously update our question bank with up-to-date questions related to fast evolving technical topics.
- By using actual Q/A threads as inspiration, we are testing for concepts that are proven to be non-trivial in the real world.
- By leveraging Q/A threads into test-questions, we achieve much lower costs to generate a question compared to employing experts.
- By continuously monitoring the Internet for leaked questions, we can quickly eliminate opportunities for cheating.

We also introduce the following novel aspects for question evaluation:

- By utilizing exogenous ability metrics, such as wages, we evaluate questions as predictors of market performance metrics.
- By continuously evaluating the test-questions we also find questions that have been leaked, since such questions suddenly lose their ability to discriminate between users with different ability levels.

The remaining article is organized as follows. First, in Section 2, we give an overview of our system and discuss the functionality of each one of its components. Then, in Section 3 we describe the process of mining questions from Stack Overflow to be used as candidate test questions for our system. Subsequently, in Section 4 we describe how we leverage Item Response Theory to evaluate the generated questions. Finally, in Section 5 we discuss our contributions and future extensions of our system.

2. System Overview

Our skill-test generation and evaluation system consists of multiple components, which are shown in Figure 2. This section provides an overview of the workflow of the system, giving brief descriptions of the functionality of each component. Some of the system components depend on human input whereas others operate automatically.

2.1. Overview

The life of a question in our system starts from extracting a Q/A thread from a Q/A-site. The question is then mapped to particular skills and evaluated with respect to its appropriateness to serve as a seed for a test question of the topic at hand. Thereafter the question is edited into a standardized test question, it is reviewed for correctness and forwarded to the pool of testing questions. There, the question collects answer-impressions from multiple users which are then used for its evaluation using Item Response Theory metrics. Depending on the outcome of the evaluation the question is rejected, re-evaluated or accepted. The accepted question metrics are used to evaluate users with respect to their expertise in a particular skill.

2.2. Question Ingestion Component

The question ingestion component of our platform is responsible for collecting new "question seeds" from online resources in order to keep the question pool wide-ranging and fresh. In particular, the Ingestion component communicates with the Q/A site and fetches question and answer threads that are then stored in a database, together with a variety of metadata. The threads are labeled then as "promising" or not by an automatic classification model (see Section 3 for details). The threads rejected by the classifier are removed from the question seed bank, whereas the accepted ones are forwarded to the editors to be transformed to standardized questions.

2.3. Question Editor

QA threads labeled as promising leave the *question ingestion component* are forwarded to the question editors. Question editors are human contractors with expertise on the topic of the test. question editors visit the Q/A thread and upon reading the question as well as the relevant answers, the editor reformulates the question so as to match the style of a test question. The editor may then also use the answers in order to generate a list of multiple-choice questions. If more than one answers are valid, the candidate taking the test may be asked to pick the best among them. If the editor does not consider a Q/A thread appropriate to be converted to a test question she directly discards it.

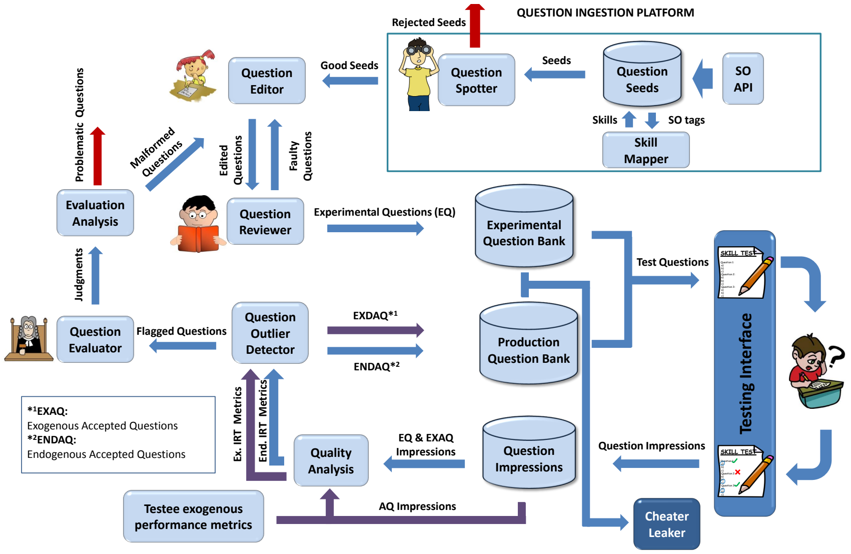


Figure 2: The architecture, components, and workflow of our platform.

Clearly, reformulating a Q/A thread into a test question is a much faster process than having an editor choose an appropriate topic and then create an appropriate question "from scratch". This makes the question generation process of our system more cost effective since domain experts, payed by the hour, can, in the same time, generate more questions when transforming existing content than creating the question from the beginning. Figure 3 illustrates a question transformation example of a Q/A thread from Stack Overflow to a java skill test question. The editors' reported time to reformulate the particular question thread was 5 minutes.
Apart from efficiency, the transformation process is also more effective since Q/A sites contain questions that actually arise in practice rather than the possibly artificial problems that an editor will think of.

2.4. *Question Reviewer*

Once the question is generated by the question editor it is forwarded to a question reviewer. The reviewer does not need to have expertise with the topic at hand, but must have a good handle of the English language. The reviewer is responsible to checks for for spelling, syntactical, or grammatical errors, and ensures that the question that is formulated follows the guidelines suggested by the test standards. The test standards include question text length, answer option count, answer text length, vocabulary usage etc. Each question that is approved by the reviewer becomes *experimental* and is committed to the experimental question bank. Non-approved questions are sent back to the question editor for re-editing.

2.5. *Question Bank: Experimental and Production*

The experimental question bank stores questions that are created by our system, but are not yet evaluated. The experimental questions are included in the tests but compose only a 10% to 20% of the test questions. Clearly the experimental questions are not being used for the evaluation of the test takers. Once the experimental questions receive a substantial number of answers, they are forwarded for evaluation to the *quality analysis* component. The production question bank stores those questions that are shown to users in tests and that are used for evaluation. Production questions are also re-evaluated periodically using the Quality Analysis component.

2.6. *Quality Analysis*

The quality analysis component is the part of the system that is responsible for computing quality metrics for each question. Its main functionality is the quality evaluation of the test questions. The experimental questions are evaluated using the "endogenous" metrics (i.e., whether the performance of the users in that question correlates well the overall test score). If a question performs well it graduates into production. The production questions are evaluated periodically using exogenous metrics (i.e., how well they can predict the market performance of the users a few months after the test). We describe the process in detail in the corresponding section below.

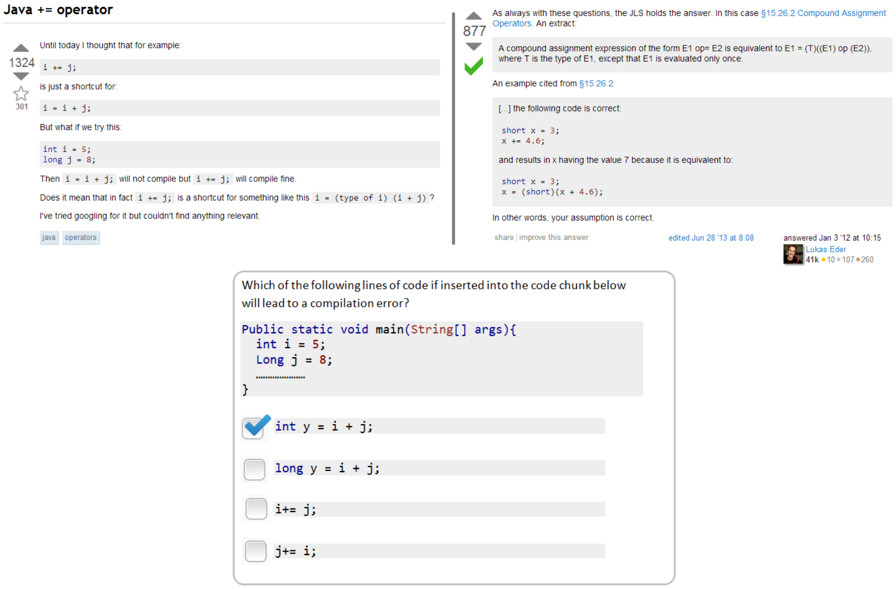
In addition to calculating the quality metrics, the quality analysis component also has an outlier detector that identifies questions that behave differently than others; such questions are forwarded to human experts that examine whether the question has any technical error, ambiguity, and so on. Problematic questions that can be corrected are edited and reintroduced in the system as experimental questions. Ambiguous and irrelevant questions are typically discarded, as they are difficult to fix. A question is also discarded if no particular problem has been identified but the question still exhibits unusual behavior. A common cause for the problematic behavior is that the question has been compromised. Even if the question is correctly formulated, and theoretically is able to discriminate test takers with different ability levels, when it has leaked, a user's answer to this question is not a reliable signal for the user's ability in the topic, leading to strange statistical behavior.

2.7. *Cheater Leaker*

The role of the cheater leaker is to identify compromised questions on the web. The "cheater leaker" component issues continuously queries against popular search engines, monitoring for leaked versions of the test questions.

The techniques that are used for detecting highly similar documents on the web are not technically novel. We extract small "unusual" n-grams from the questions and feed these as phrase queries in search engine APIs, to detect pages with similar content. We use both existing commercial services for approximate querying (e.g., CopyScape) and "query by document" techniques proposed in Yang et.al [21].

The main goal of the cheater leaker is to prevent test takers from searching the question or part of a question online and directly finding the correct answer option in certain forums. If the question was not reformulated to be significantly different from the original that was found in the Q/A thread, or is still similar to its older version



Java += operator

Until today I thought that for example:

```
i += j;
```

is just a shortcut for:

```
i = i + j;
```

But what if we try this:

```
int i = 5;
long j = 8;
```

Then `i = i + j;` will not compile but `i += j;` will compile fine.

Does it mean that in fact `i += j;` is a shortcut for something like this `i = (type of i) (i + j)` ?

I've tried googling for it but couldn't find anything relevant.

As always with these questions, the JLS holds the answer. In this case §15.26.2 Compound Assignment Operators. An extract:

> A compound assignment expression of the form E1 op= E2 is equivalent to E1 = (T)((E1) op (E2)), where T is the type of E1, except that E1 is evaluated only once.

An example cited from §15.26.2

> [...] the following code is correct:
>
> ```
> short x = 3;
> x += 4.6;
> ```
>
> and results in x having the value 7 because it is equivalent to:
>
> ```
> short x = 3;
> x = (short)(x + 4.6);
> ```

In other words, your assumption is correct.

edited Jun 28 '13 at 8:08

answered Jan 3 '12 at 10:15
Lukas Eder
41k ●10 ●107 ●260

Which of the following lines of code if inserted into the code chunk below will lead to a compilation error?

```
Public static void main(String[] args){
  int i = 5;
  Long j = 8;
  ......................
}
```

☑ `int y = i + j;`

□ `long y = i + j;`

□ `i+= j;`

□ `j+= i;`

Figure 3: Example of Q/A Thread (top) transformation to a multiple-choice Java test question (bottom).

that had been leaked, this is detected by the cheater leaker. People taking a test face a time constraint of slightly more than one minute per question on average. Hence, a test taker can take advantage of a leak only if a) she can locate it very quickly and b) she can directly interpret the answer that she sees online into the appropriate answer in the test. Even if a question has been leaked and cannot be identified by the cheater leaker, but workers somehow are able locate and use the leaked answers consistently, this will be identified in the long run by the question evaluation component since the question discrimination will gradually decrease, especially for the exogenous metrics of ability.

Once a question is located "in the wild," a human visits the identified web site and examines whether indeed it contains the question and the answers. A question is then marked as "leaked" and gets retired from the system: the leaked questions are then released as practice questions and teaching/homework material for learning the skill. This component is also used to ensure that when the question is originally created by the editor, it is sufficiently reworded to avoid being located by simple web queries. CheaterLeaker inspects questions in the editing phase and warns question editors if the source documents can be detected on the web through querying.

We do not check for similarity with existing questions, although this is a good feature to incorporate for ensuring non-overlap of questions. At this point we rely on the Stack Overflow to avoid duplicate question generation. Stack Overflow tags questions that are found to be near duplicate versions of other questions with a "duplicated" tag.

3. Question Generation Process

Aiming to continuously generate new and up-to-date questions, our system leverages content that is available in question answering sites, to generate seeds for new test questions. In this section, we describe how our system uses "crowdsourced" content in question-answering website, such as Stack Exchange, in order to create seed ideas for the generation of test questions. Since not all question-answer threads are suitable for test questions, we also describe how we build an automated algorithm for identifying "good" threads, which makes the life of question editors easier.

3.1. Stack Exchange

Stack Exchange (SE) is network of more than a hundred sites with question answer threads on different areas ranging from software programming questions to Japanese language and photography questions. SE has an available API that provides programmatic access for downloading questions threads that are posted on these platforms. Along with a question text and title the API also extracts all the answers and comments associated with the question as well as a number of other semantically rich question, answer and comment features. These semantically rich features include the question view count, the up-votes and down-votes that the question and its answers have received, the question, answer, and comment author reputation scores, the tags associated with the question and many more.

Our current system focuses on testing for technical-skills and therefore we leverage the content on Stack Overflow. Stack Overflow is Stack Exchange's most popular site and it defines itself as "a question and answer site for professional and enthusiast programmers"[5]. It has more than three million subscribed users and more than six million questions associated with $35K$ topics (tags). More than 91% of the questions on Stack Overflow have at least one answer.

Each question on Stack Overflow is associated with one ore more tags (topics). Our system translates these tags into topics for which it generates tests. This way each question can be directly categorized into a test-area. Table 1 shows the 10 most popular topics which compose slightly more than 20% of the total volume of questions[6].

Although the volume of the available questions in sites such as Stack Overflow provides us a large bank of candidate question seeds, only a subset of the Q/A threads are suitable for the generation of test questions and we need to identify the most promising threads to avoid overwhelming the editors with false leads.

Needless to say, it is not feasible or desirable to manually examine all threads to examine which threads are the most promising for generating test questions. Ultimately, we want to automate the process of identifying good threads and then use them as seeds for question generation. Ideally, the question should test something that is confusing to users when they learn a skill, but clear for experts.

[5] `http://stackoverflow.com/`

[6] This data comes from a Stack Overflow dump in January 2014

Topic	Questions	Percentile (%)
C#	508,194	3.08
Java	468,554	2.84
PHP	433,801	2.63
Javascript	433,707	2.63
Android	377,031	2.29
Jquery	355,800	2.16
C++	222,599	1.35
Python	216,924	1.32
HTML	198,028	1.20
mysql	184,382	1.12

Table 1: Top-10 popular Stack Overflow tags

3.2. Question Spotter

Towards this goal we trained a classifier that we call Question Spotter. The role of the question spotter is to identify promising Q/A threads in order to reduce the load of threads that have to be processed by the question editor. For a thread to be a promising it has at least to have the following properties:

- The question and answer sizes should not be too large to process. The question editor needs to be able to quickly read and understand the question, decide whether it is appropriate to test a particular skill and transform it into a test question. There are questions on Stack Overflow with more three thousands characters of free text and code snippets which would require more than ten minutes to just read the question.

- The question has to be relevant to the general topic at hand. It should not require the expertise of the test taker in a particular sub-domain of the topic. For instance, when examining java a test question should not require the test taker to know a particular Java platform like Spring. Moreover the question should not require the knowledge of some very specific and useless detail of the topic that is tested. An example found on Stack Overflow is a Q/A thread discussing an error of Java 5 when subtracting two timestamps that was caused by a 5 minutes and 52 seconds setback of clocks in Shanghai at midnight of December 31^{st} 1927.

- The question hast to have at least one answer on Stack Overflow. The editor has to be inspired not only be the question but also by the answers that users give to the question. Even better if a Q/A thread has multiple answers that can give the editor a better idea what is confusing in the topic and write a more challenging but less ambigous test-question.

- It has to be a strictly technical question and not any topic-related question. A counter example that can be found on Stack Overflow is the question “The Definitive C++ Book Guide and List” tagged with the tag C++. This is also one of the most popular (in terms of upvotes) questions on Stack Overflow.

- The question has to have a clear correct answer and not be ambiguous. Example of ambiguous questions are questions of the form ”What is the best way to do x?” where the term “best” is not clearly defined. Usually such questions do not have any accepted answer coming with the question.

In order to train the question spotter, we follow a labeling of Q/A threads as good or not. A test question is always judged with respect to a particular skill. Using the labels of the question, we then build a classification model that assigns automatically a label to each incoming QA thread. For the Java topic we sampled and labeled one thousand questions as “good” or “bad” to serve as test-questions. About 35% of them were labeled as good question seeds, whereas the rest as bad ones. By measuring the correlation of each feature with the question's label, we found that the most informative features about the appropriateness of a Q/A to serve as a question seed were the following: (1) the length of the question text (in number of characters), (2) the popularity (in number of questions associated with a tag) entropy of the tags associated with question, (3) the number of tags associated with the question, (4) the average question score (number of upvotes-number of downvotes) per week, (5) the entropy upvotes that the question answers have received, (6) the average weekly answer score (upvotes-downvotes), (7) the weekly number of views of the question, (8) the number of answers given to the question, and (9) the maximum of all reputations of the users that answered the question. The last five features were positively correlated with “good” questions whereas the first four were negatively correlated.

We used these labeled questions, and the tree package of R[7] to train a classification decision tree[22].[8] Our objective was to optimize for the precision of the results and minimize the number of false positives in the results (i.e., minimize the bad threads listed as good). For a recall of 90% we achieved precision of 75%. The main goal of the spotter is to reduce the number of not-promissing threads that will be processed and rejected by the question editor.

We also performed a qualitative assessment of the features used, to get a better understanding of what makes a Q/A thread a good seed for a test question. We noticed that a large number of upvotes is actually a *negative* predictor for suitability for the thread to generate a good test questions: highly voted questions tend to ask about arcane topics with little practical value; on the contrary, threads with a large number of answers and high-entropy distribution of upvotes across the answers, signal the existence of a topic that is confusing users, with many answers that can serve as “distractor answers” [23]. We also found that question threads frequently visited by many users indicate questions on common problems for a variety of expertise levels for the topic at hand. Moreover, we found that questions that had been answered by at least one user with high reputation tend to be very interesting and tricky problems. Finally, we found that Q/A threads tagged with multiple popular tags are not directly related to one specific topic and are therefore less likely to be appropriate candidates for test-questions.

[7] `http://cran.r-project.org/web/packages/tree/tree.pdf`

[8] Decision trees are tree-like classification models. Each node in the tree evaluates the value of a particular feature and decides to which of its children the incoming data point will be forwarded. In the end of the path the leaf nodes of the tree assign the classification label to the incoming data point.

Of course, the true question is not the predictive ability of the question spotter component, but rather how many of the questions inspired by the seeds ended up being good test questions. We discuss that topic in the next section.

4. Question Quality Evaluation

Our system can scalably generate a large number of questions for skill testing. However, the objective is not just to generate a large number of questions, but to generate a large number of *good* questions. This section discusses how our question evaluation component works. The question analysis component generates a set of metrics to evaluate the quality of the questions in the question banks. We compute these metrics using standard methods from *item-response-theory (IRT)*. IRT is a field of psychometrics employed for evaluating the quality of tests and surveys that measure abilities, attitudes, and so on. The prerequisite for analyzing a question (an "Item" in IRT) is for the question to be answered by a sufficiently large number of test takers. Once we have that data, IRT can then be used to examine how well the test question measures the "ability" θ of a test taker. Traditionally, the θ is approximated by the score of the user in the overall test, and is rather "endogenous." As a key contribution of our system, in addition to the endogenous measure of ability, we also use "exogenous" market performance metrics for measuring the ability θ of a test taker as demonstrated in the market, and not just based on the test results.

4.1. Basics of Item Response Theory

Before describing our question evaluation process in detail, we briefly discuss some preliminaries on Item Response Theory [18]. The first assumption in IRT is that the test takers have a single ability parameter θ, which represents the subject's ability level in a particular field, which customary, we consider to have a $N(0, 1)$ normal distribution, with the population mean having $\theta = 0$. The second assumption is that items are conditionally independent, given an individual's ability θ.[9] Given these two assumptions, the basic concept of IRT is that each question can be characterized by the probability $P(\theta)$ that a user with an ability θ will give a successful answer to the question. This function $P(\theta)$ is called Item Characteristic Curve (ICC) or item-response-function (IRF) and has the following general form:

$$P(\theta) = c + \frac{d - c}{1 + e^{-a(\theta - b)}} \tag{1}$$

The parameter a is called *discrimination* and quantifies how well the question discriminates between test takers with different ability levels; higher values of a result in a steeper curve, which means that the probability of answering correctly increases sharply with the ability of the test taker. The parameter b is called *difficulty*; it corresponds to the value of θ where $P(\theta) = 0.5$ and is also the inflection point of the curve;

[9] Conditional independence means that the probability of a worker generating a correct answer to a question q depends only in the ability of the user θ and not on whether the user answered correctly (or incorrectly) other question q'. Formally: $P(q|\theta, q') = P(q|\theta)$.

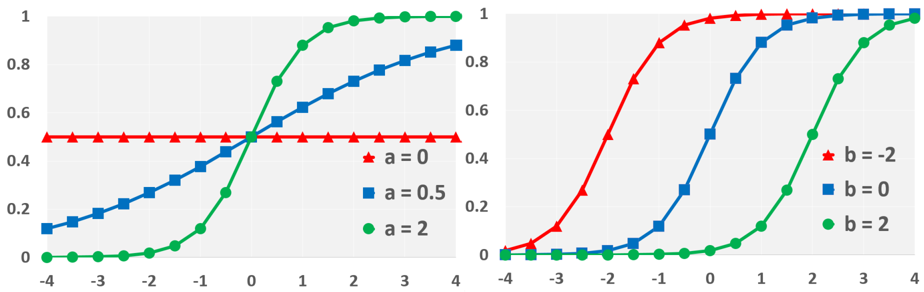


Figure 4: Illustrations of the 2PL "item characteristic curve" for different discrimination (left) and difficulty (right) values. X-axis denotes the test taker ability θ and Y-axis probability $P(\theta)$.

higher values mean that only high ability test takers answer the question correctly. Finally, c is the probability of guessing the correct answer randomly for each question and d the highest possible probability of answering a question correctly.

Figure 4 illustrates how the ICC changes for different values of discrimination and difficulty. On the left, the question's difficulty is set to zero and the lines show the ICC for three discrimination values. When the discrimination is zero, the line is flat and it is obvious that there is no correlation between the test taker ability and the probability of answering the question correctly. On the right plot of the figure, the question's discrimination is set to 2 and the three lines show the ICC for three difficulty values. Smaller difficulty values shift the steep part of the curve to the left, and let test takers with lower ability levels to have better chances of answering the question correctly.

An important additional metric to consider is the *Fisher information* $I(\theta)$ of the $P(\theta)$ distribution [24, Section 2.3.1]. In our context, the Fisher information of a question quantifies how accurately we can measure the ability θ (the unknown parameter) for a user after observing the answer to the question (the observed random variable). Formally:

$$I(\theta) = a^2 \frac{e^{-a(\theta-b)}}{(1+e^{-a(\theta-b)})^2} \tag{2}$$

In general, highly discriminating items have tall, narrow information functions and they can measure with accuracy the θ value but over a narrow range. Less discriminating questions provide less information but over a wider range. Intuitively, highly discriminative questions can provide a lot of information about the ability of a user around the inflection point (as they separate the test takers well) but are not providing much information in the flatter regions of the curve.

An important and useful property of Fisher information is its additivity. The Fisher information of a test is the sum of the information of all the questions in the test. So, when creating a test, we can select questions with that have high $I(\theta)$ across a variety of θ values to be able to measure well the ability θ across a variety of values. Of course, if we want to measure more accurately some regions, we can add more questions that have high $I(\theta)$ for the regions of interest.[10]

4.2. Question Analysis based on Endogenous Metrics

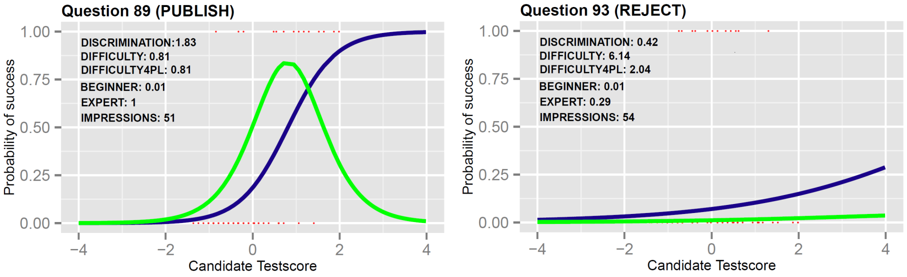


Figure 5: ICC (blue) and information curves (green) for an accepted (left) vs. a rejected(right) experimental question.

Following the paradigm of traditional IRT, our first quality analysis uses as a measure of the ability θ the test score of the test taker, computed over only the the production questions in the test (and not the experimental). As described in subesction 2.5 production questions are the questions in the test that are used for the test taker evaluation whereas experimental questions are shown to users in small portions within a test but are not used for their evaluation.

The raw test score for each user i is then converted into a normalized value θ_i, so that the distribution of scores is a standard normal distribution. Instead of allowing all questions to contribute equally to the raw score, some IRT algorithms allow each question to contribute differently to the score, according to the discrimination power and the difficulty. Although more principled, the changes in the scores are often negligible with more than 95% of the scores remaining the same and with the additional problem that it is not possible to explain the scoring mechanism to the test takers. Once we have the ability scores θ_i for each user i, we then analyze each question j. The answer of the user in each question is binary, either correct or incorrect. Using the data, we fit the ICC curve and we estimate the discrimination a_j and the difficulty b_j for each question.

For an experimental question to move to production, we require the discrimination to in the top-90% percentile across all questions (i.e., we reject the bottom-10% of the questions, when ranked based on their discrimination value). We also require discrimination to be positive; when discrimination is negative, the question's ICC curve is decreasing as ability θ increases. Hence, as the ability of users increases, their probability to answer the question correctly *decreases*. Such negative discrimination values typically indicate that there is something wrong with the question. In our study we

[10] Typically, we want to measure accurately the ability of the top performers while we are rather indifferent when separating the bottom-50%. Unfortunately, in reality, it is difficult to construct many test questions that have *both* high discrimination *and* high difficulty.

found that questions with negative discrimination either were written in a very misleading way, or had accidentally the wrong answer marked as correct by their creator.

Figure 5 shows the ICC and information curves for two questions. An accepted question has a high discrimination value, and correspondingly high Fisher information; a rejected question typically has low discrimination and low Fisher information. When analyzing existing tests, we also observed questions with high but *negative* discrimination values; these questions almost always had an incorrect answer marked as correct, or were "trick" questions testing very arcane parts of the language. Figures 6 and 7 show the ICC and information curves of two questions about Java. The blue (logistic) curve illustrates the ICC curve and the green curve the information curve. Figure 6 shows a question with high discrimination and medium difficulty, whereas Figure 7 shows a question with high difficulty and low discrimination.

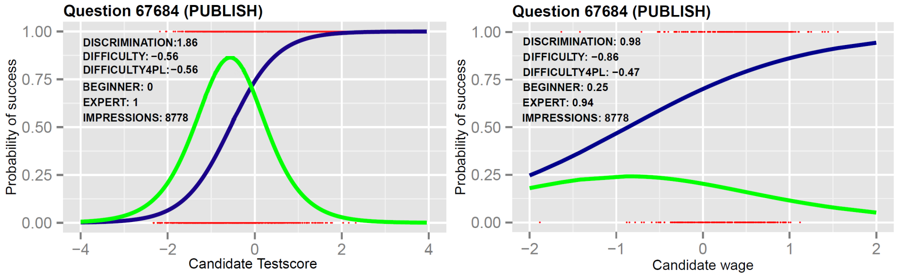


Figure 6: Example of accepted production question analysis endogenous (left) vs. exogenous (right) metrics.

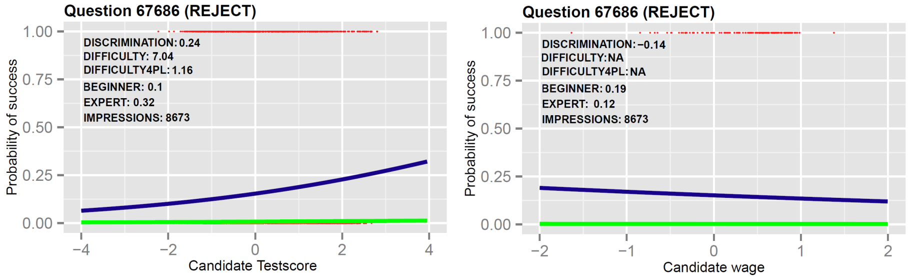


Figure 7: Example of rejected production question analysis endogenous (left) vs. exogenous (right) metrics.

4.3. Question Analysis based on Exogenous Metrics

A common complaint about tests is that they do not focus on topics that are important "in the real world" [13, 14, 15, 16, 17]. As an important contribution of our work, we decided to also use "exogenous" ability metrics to represent the test taker θs. Exogenous ability metrics measure the success of the test taker in the labor market, as opposed to the success while taking the test. Examples of these metrics are the test takers average wage, her hiring rate, the jobs that he has completed successfully etc. Using exogenous metrics makes the evaluation of the questions more robust to cheating, and can indicate easier which of the skills tested by the question are also important in the marketplace. In this work, we focus on worker's wage because it is a more objective metric regarding a worker's ability. The hiring rate metric depends on whether the tasks assigned to a worker are long-term or short-term and depend on how frequently the worker applies for employment. The same is true for the "jobs completed" metric. Wages are not affected by the duration of the tasks, or by the preferences of the workers for working with multiple or with a single employer. Hence, we present the results using the log of wages three months after the test, to represent the test taker's ability θ.

Not surprisingly, the questions do not exhibit the same degree of correlation with the exogenous user abilities compared to the endogenous ability (the user test score itself). Tests are composed of questions and therefore the test score is directly related to the score of a test taker to an individual question in the test. On the other hand, all exogenous metrics include many other aspects besides the knowledge in a particular area. Examples of such aspects include efficiency in development, project planning, responsible personality etc. Figure 6(bottom) shows the ICC and information curves of the same question as the top plot but computed using the exogenous ability metrics. We observe that the discrimination of the question that was computed using the exogenous ability metrics is relatively high (0.98) discrimination but still not as high as the the discrimination computed using the endogenous metrics (1.86). The same holds for the two plots in Figure 7. Both plots show a low quality question, with the discrimination computed by the exogenous ability metrics actually being negative. The pattern holds across all questions that we have examined. One immediate, practical implication is that we need more test takers to be able to estimate robustly the discrimination and difficulty parameters for each question.

Our analysis with an exogenous ability metric has two objectives. First, we better understand the contractors and their ability to perform well in the marketplace. Second, we also determine which of the test questions are still useful for contractor evaluation: for questions that are leaked, or questions that are now outdated (e.g., deprecated features) the exogenous evaluation shows a drop of discrimination over time, giving us signals that the question has to be removed or corrected.

4.4. Experimental Evaluation

Our approach for generating tests from Q/A sites has the clear advantages of being able to generate new questions quickly, compared to the existing practice of using a "static" pool of test questions. However, there are two key questions when considering this approach: (a) How do the questions perform compared to currently used test questions, and (b) What is the cost for generating these questions?

In order to evaluate the benefit of our system compared to the existing approach of using a static question bank composed of licensed questions, we generated test questions with our platform for the following skills: PHP, Python, Ruby on Rails, CSS, HTML, and Java. For each test we had 50 questions. We then used the skill testing interface of oDesk that allowed us to collect responses to our questions by injecting a small number of them at-a-time to the oDesk skill-tests. Our questions were not used for the oDesk user evaluation but we collected at least 50 responses for each. We also

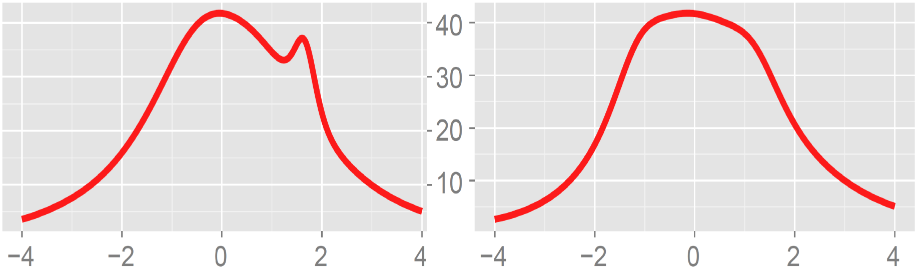


Figure 8: Information curves for test with questions generated by domain experts (left), vs. new test with questions inspired by Stack Overflow threads and generated by our system.

had access to the exogenous metrics of oDesk users to evaluate our methods. Hence, for each skill we had the existing test that contained questions from a "static" question bank, generated by domain experts, and the new test, which contained only questions generated by our system, using Stack Overflow threads.

For each of these two tests (tests with static bank questions, and test composed of questions generated by our platform) both composed of same number of questions, we computed the information curve for the test, by summing the information gain of all its questions. Figure 8 displays the results for the Java test. The left plot shows the information curve for the test containing the "static question bank" questions; the right the information gain for the test containing questions generated by our platform. The x-axis is the ability level of the test takers and the y-axis the information of the test for the particular ability level; as a reminder, high information values mean higher precision of the test when measuring the ability of a worker with a certain ability. Both tests behave similarly, indicating that our questions have the same quality on average as the questions that are generated by domain experts.

We also examined how many of the questions in the two tests were able to pass the evaluation that used the exogenous ability (wage) as the ability metric. When evaluating the domain expert questions, 87% of the questions were accepted, whereas the questions that our platform generated have a 89% acceptance rate. The numbers are roughly equivalent, indicating that our platform can generate questions at the same level of quality (or even higher) than the existing solutions.

Given that the quality of the our tests is equivalent to the existing tests that we can acquire from a question bank, the next question is whether it makes financial sense to create questions using our platform. The cost of the question generated by our platform ranged from \$3/question to \$5/question, depending on the skill tested, with an average cost of \$4/ question. For the domain-expert questions, the cost per question was either a variable \$0.25/question *per user taking the test* or \$10 to buy the question[11] Therefore, it is also financially preferable to use our platform to generate questions compared to

[11] The numbers correspond to \$10 per user taking a 40-question test, or \$500 to buy the full question bank that contained 50 questions.

using existing question banks; in addition to being cheaper, our platform also allows for a continuous refreshing of the question bank, and allows the retired questions to be used by current users as practice questions for improving their skills.

5. Discussion and Conclusions

We presented a scalable testing and evaluation platform. Our platforrm leverages content from user-generated question answering websites to continuously generate test questions, allowing the tests to be always "fresh" minimizing the problem of question leakage that unavoidably leads to cheating. We also show how to leverage item-response-theory to perform quality control on the generated questions and, furthermore, we use marketplace-derived metrics to evaluate the ability of test questions to assess and predict the performance of contractors in the marketplace, making it even more difficult for cheating to have an actual effect in the results of the tests.

One important direction for the future, is to build tests that have higher discrimination power for the top-ranked users than for the low-ranked ones (e.g., discriminate better between the top-5% and top-20%, compared to between the bottom-5% and bottom-20%). We expect the use of adaptive testing to be useful in that respect, as we can have tests that terminate early for the low-ranked users, while for the top-ranked users, we may ask more questions, until reaching the desired level of measurement accuracy. Also, we want to apply STEP for generating tests for for non-programming skills by leveraging non-technical Q/A sites, and even generate tests for MOOCs by analyzing the contents of the discussion boards, where students ask questions about the content of the course, the homeworks, etc. We believe that such a methodology will allow the tests to be more tailored to the student population and that can measure better the skills that are expected in the marketplace.

References

[1] T. Hossfeld, M. Seufert, M. Hirth, T. Zinner, P. Tran-Gia, R. Schatz, Quantification of youtube qoe via crowdsourcing, in: Multimedia (ISM), 2011 IEEE International Symposium on, 2011, pp. 494–499.

[2] T. Hossfeld, C. Keimel, M. Hirth, B. Gardlo, J. Habigt, K. Diepold, P. Tran-Gia, Best practices for qoe crowdtesting: Qoe assessment with crowdsourcing, Multimedia, IEEE Transactions on 16 (2) (2014) 541–558.

[3] A. Davies, D. Fidler, M. Gorbis, Future work skills 2020, Institute for the Future for University of Phoenix Research Institute, 2011.

[4] P. Resnick, K. Kuwabara, R. Zeckhauser, E. Friedman, Reputation systems, Communications of the ACM 43 (12) (2000) 45–48.

[5] C. Dellarocas, The digitization of word of mouth: Promise and challenges of online feedback mechanisms, Management Science 49 (10) (2003) 1407–1424.

[6] H. Zhu, S. P. Dow, R. E. Kraut, A. Kittur, Reviewing versus doing: Learning and performance in crowd assessment, in: Proceedings of the 17th ACM Conference on Computer Supported Cooperative Work &#38; Social Computing, New York, NY, USA, 2014, pp. 1445–1455.

[7] S. Dow, A. Kulkarni, S. Klemmer, B. Hartmann, Shepherding the crowd yields better work, in: Proceedings of the ACM 2012 Conference on Computer Supported Cooperative Work, 2012, pp. 1013–1022.

[8] C. Kulkarni, K. P. Wei, H. Le, D. Chia, K. Papadopoulos, J. Cheng, D. Koller, S. R. Klemmer, Peer and self assessment in massive online classes, ACM Trans. Comput.-Hum. Interact. 20 (6) (2013) 33:1–33:31.

[9] A. Pallais, Inefficient hiring in entry-level labor markets, Working Paper 18917, National Bureau of Economic Research (March 2013).

[10] G. A. Akerlof, The market for ”lemons”: Quality uncertainty and the market mechanism, The Quarterly Journal of Economics 84 (3) (1970) 488–500.

[11] M. Spence, Job market signaling, The quarterly journal of Economics 87 (3) (1973) 355–374.

[12] M. S. Wingersky, L. L. Cook, Specifying the characteristics of linking items used for item response theory item calibration, Educational Testing Service, 1987.

[13] W. J. Popham, Why standardized tests don't measure educational quality, Educational Leadership 56 (1999) 8–16.

[14] A. R. Jensen, Bias in mental testing., ERIC, 1980.

[15] F. M. Newmann, A. S. Bryk, J. K. Nagaoka, Authentic intellectual work and standardized tests: Conflict or coexistence?, Consortium on Chicago School Research Chicago, 2001.

[16] S. Geiser, M. V. Santelices, Validity of high-school grades in predicting student success beyond the freshman year: High-school record vs. standardized tests as indicators of four-year college outcomes, Tech. rep., University of California–Berkeley (2007).

[17] J. Fleming, N. Garcia, Are standardized tests fair to african americans?: Predictive validity of the SAT in black and white institutions, Journal of Higher Education 69 (5) (1998) 471–495.

[18] H. J. R. Ronald K. Hambleton, Hariharan Swaminathan, Fundamentals of Item Response Theory, 3rd Edition, SAGE Publications, 1991.

[19] S. E. Embretson, S. P. Reise, Item response theory, Psychology Press, 2000.

[20] G. Bergersen, D. Sjoberg, T. Dyba, Construction and validation of an instrument for measuring programming skill, Software Engineering, IEEE Transactions on 40 (12) (2014) 1163–1184.

[21] Y. Yang, N. Bansal, W. Dakka, P. G. Ipeirotis, N. Koudas, D. Papadias, Query by document, in: Proceedings of the Second International Conference on Web Search and Web Data Mining, WSDM 2009, Barcelona, Spain, February 9-11, 2009, 2009, pp. 34–43.

[22] J. R. Quinlan, Induction of decision trees, Machine learning 1 (1) (1986) 81–106.

[23] L. Guttman, I. Schlesinger, Systematic construction of distractors for ability and achievement test items., Educational and Psychological Measurement 27 (3) (1967) 569–580.

[24] M. J. Schervish, Theory of statistics, Springer, 1995.