# Seeing the Trees for the Wood: Holistic Interpretation of a Policy for Quality Control

**************
**********
**@**

**Abstract**

One of the most challenging issues for online courseware engineering is to maintain the quality of instructional components, such as written text, video, and assessments. However, it is hard to know how instructional components on particular online courseware contribute to students' learning. To address this challenge, we propose an innovative application of reinforcement learning (RL). A collection of actions suggested as a policy is *holistically* analyzed to evaluate the usefulness of those actions with regards to achieving ideal outcomes. The proposed RL application is invented for human-in-the-loop learning engineering method called RAFINE. In the RAFINE framework, a machine generates a list of the least contributing instructional components on the given online courseware based on the *holistic interpretation* of a policy, and courseware developers refine those suggested components. As a proof of concept, this paper describes an evaluation study where online learning was simulated on hypothetical online courseware. The results showed that over 90% of ineffective instructional components were correctly identified as ineffective on average.

## Introduction

With the rapidly growing popularity of online courses, there has been a heavy demand for practical engineering methods to efficiently create effective online courseware (Shapiro et al. 2017). Even though some well-known design principles for e-learning technologies are available (e.g., Dede, Richards and Saxberg 2018; Guàrdia, Maina and Sangrà 2013), such principle-based approaches still require labor-intensive iterative engineering for practical courseware development at scale (Fishman et al. 2004).

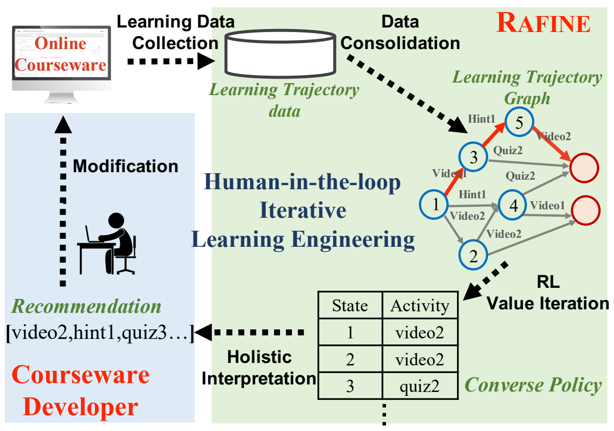
One of the challenges in the principle-based iterative system development is to identify a spot of repair—i.e., understanding a cause of undesired system performance. When creating online courseware, an initial version of the courseware designed based on existing learning principles will be used with actual students while collecting interaction and learning outcome data. Experienced learning engineers then analyze the data to identify issues with the courseware. The learning curve analysis (Martin et al. 2011) and the assessment items analysis (Milligan and Griffin 2016) are the two most commonly used analytic techniques. However, these techniques only apply to assessment items while other types of instructional components such as video clips and written texts must also be included in the analysis.

A practical problem that we aim to solve is to automatically identify relatively less effective instructional components on existing online courseware based on actual students' learning data. To solve this problem, we propose a unique application of reinforcement learning (RL) as a tool to analyze the degree of contribution of individual instructional components to students' learning.

The proposed application of RL is a building block of a method for evidence-based, human-in-the-loop, iterative learning engineering that we call RAFINE (Reinforcement learning Application For INcremental courseware Engineering). Figure 1 shows how humans and AI collaborate to iteratively improve the quality of courseware. RAFINE generates a recommendation for courseware refinement based on the holistic interpretation of a policy. The courseware developer then refines the courseware by modifying the ineffective instructional components listed in the recommendation.

We assume that each student's learning trajectory on the given online courseware is converted into a state transition graph. The all students' state transition graphs are then consolidated into a single Markov decision process (MDP) where states represent students' intermediate learning status and actions represent instructional components taken that changed learning status. We then propose a variant version of value iteration technique commonly used for RL to compute a policy representing the *least* optimal actions

Copyright © 2020, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.



| State | Activity |
|---|---|
| 1 | video2 |
| 2 | video2 |
| 3 | quiz2 |

Figure 1 : Overview of the RAFINE method

to be taken that we call a *converse policy*.

Our central hypothesis is that a collection of actions suggested as a converse policy over the all states in a given MDP can be used to infer the quality of instructional components (i.e., actions) actually taken by students on the target online courseware.

The originality of the proposed work is rooted in the way that the policy is interpreted. The traditional application of RL mostly relies on an *atomic interpretation* of the policy. The policy, $\pi(s)$, as a function of the state is used to predict an optimal action to be taken at a certain state $s$.

We propose a *holistic interpretation* of a converse policy to analyze the usefulness of available actions. Let $\mathcal{A}^*$ be a collection of the least optimal actions under a given converse policy, i.e., $\mathcal{A}^* = \{a_{s_i} | \pi(s_i) = a_{s_i}, s_i \in \boldsymbol{S}\}$ where $\boldsymbol{S}$ is a set of all states in a given MDP and $a_{s_i}$ is an action chosen by the converse policy $\pi$ at state $s_i$. For the sake of explanation, we call $a_{s_i}$ a *policy action* and $\mathcal{A}^*$ a *policy action set*. Note that by definition, $\mathcal{A}^*$ contains duplicated policy actions hence $|\mathcal{A}^*| = |\boldsymbol{S}|$.

We argue that by holistically analyzing a policy action set, relatively ineffective actions can be identified. In the current application, this idea corresponds to evaluating the effectiveness of instructional components by holistically interpreting a policy action set of a converse policy induced from given learning trajectory data.

The goal of the current paper is to provide a proof of concept of the RAFINE method. To achieve this goal, we employed a simulation technique to generate hypothetical learning data. In the evaluation study, the RAFINE method correctly identified over 90% of ineffective instructional components on the given mock online courseware.

### Contributions of the current work

The primary contributions of the current work are essentially theoretical yet with potential for practical use: (1) We innovated a technique to holistically interpret a converse policy as a tool to detect a relative weakness among available actions with regard to achieving ideal goals. (2) As an application of the holistic interpretation of a converse policy, we proposed an evidence-based learning engineering method, RAFINE, as a building block for online courseware engineering with a broader range of instructional components as the subject of refinement. (3) We provided a proof-of-concept of the proposed RAFINE method.

For the rest of the paper, we first discuss related works regarding applications of RL for pedagogical decision making and applications of learning analytics for learning engineering. We will then introduce the RAFINE method followed by an evaluation study to address our research questions using simulated leaning data.

## Related Work

### RL applications for pedagogical decision making

RL has been applied to behavior optimization for pedagogical decision making. Most of the educational applications of RL employ an atomic interpretation of a policy to determine the optimal pedagogical strategies to provide adaptive instruction. When a pedagogical agent makes a decision, it invokes a policy function as a "mapping from perceived states of the environment to actions to be taken" (Sutton and Barto 1998) to induce an optimal action.

Azizsoltani et al. (2019) applied RL for an intelligent tutoring system to adaptively propose problems that follow different pedagogies—worked example vs. problem solving. Wang et al. (2017) applied Deep RL to tailor players' story in an educational game. The induced policy was used to provide different event sequences depending on the player's interaction logs. Rafferty et al. (2015) formulated a teaching process using POMDP and applied RL to optimize teaching actions by minimizing the time on task.

Apart from computing optimal pedagogical strategies, Zhang et al. (2019) proposed hierarchical reinforcement learning to optimize individual students' course histories to make better recommendations. Yet, the policy was used to induce an optimal sequence of "actions."

Just like the previous works described here, RAFINE induces the policy from students' learning data. Unlike these works, however, RAFINE holistically analyzes a set of policy actions across all states to identify actions (i.e., instructional components) that had relatively less contribution to students' learning.

### Learning analytics for learning engineering

For a research on the application of learning analytics for iterative courseware engineering, a common assumption is that the effect of instruction depends on the validity of an underlying skill model. The skill model (aka knowledge component model) defines a unit of analysis for pedagogical diagnostics (Koedinger, Corbett and Perfetti 2012). Learning Factor Analysis, for example, exhaustively searches through a hypothesis space to find a revised skill model that predicts students' performance better than an original skill model (Cen, Koedinger and Junker 2006). Apart from analyzing a skill model, Milligan et al. (2016) applied the item response theory to validate a proposed student model (that represents individual students' proficiency) based on students' performance on an existing online course.

Those studies have only focused on assessments. As stated for our second contribution, the RAFINE method induces a recommendation for refinement that includes *broader range of instructional components*.

## Overview of the RAFINE Method

As shown in Figure 1, an initial version of the target online courseware is used by actual students and their activities are logged. These activity data contain the standard clickstream data and students' responses for formative assessments (tagged with the correctness). Since students' activity data show a chronological record of their behavior on the online courseware, we call them the *learning trajectory data* hereafter.

The RAFINE method first consolidates individual students' learning trajectories into a single state transition graph, called a *learning trajectory graph* (LTG). The LTG is then annotated with predefined rewards. Finally, a value iteration technique is applied to compute a *converse policy* that shows the *worst* activity to be taken at each state relative to achieving the expected learning outcomes. As a consequence, the converse policy indicates which instructional components had the least contribution to students' learning at each state *in the given LTG*.

To actually identify ineffective instructional components from an induced converse policy, the collection of policy actions over the all states (i.e., the converse policy actions set) is interpreted as a whole based on a frequency—actions that frequently appear in the converse policy action set will be included in a *recommendation* as culprit for poor performance.

The RAFINE method can be iteratively applied to the revised courseware by collecting a new batch of learning trajectory data to further improve the courseware.

## Technical Details of RAFINE

The unit of analysis of the RAFINE method is an *instructional component* that constitutes online courseware. In the current study, we deal with three types of instructional components: (1) videos, (2) quizzes (aka formative assessments), and (3) hint messages associated with quizzes (we assume that all quizzes are equipped with hint messages that scaffold students to answer). Other types of instructional components (i.e., written paragraphs, tables, and figures) could also be analyzed. However, accurately tracking how students review these instructional components while learning is not straightforward—e.g., the ordinal clickstream data do not convey whether a student was reading a text instruction or not.

### Model Representation

Let $\Phi$ be a set of instructional components appearing in the given learning trajectories. Let a *learning activity* $a_i^T$ be an instructional component taken by student $i$ at time $T$ (e.g., watching a video or answering a quiz). Let $LT_i$ be a *learning trajectory* for student $i$ who has $n_i$ learning activities. $LT_i$ is a chronological record of learning activities:

$$LT_i = \{a_i^1, \ldots, a_i^{n_i} | a_i^T \in \Phi,\ T = 1, \ldots, n_i\}.$$

We assume a presence of a *skill model* that contains a set of *skills* each representing a unit of knowledge that students have to learn, aka knowledge components (Koedinger et al. 2012). This assumption implies that each instructional component is tagged with a single skill in a given skill model.

Let $\Phi^\varphi$ be a set of instructional components for skill $\varphi$. Let $LT_i^{\varphi}$ be the student $i$'s learning trajectory that contains learning activities about skill $\varphi$. A single application of the RAFINE method to $LT_i^{\varphi}$ identifies ineffective instructional components in $\Phi^\varphi$. *The RAFINE method must be applied to each set of $LT_i^{\varphi}$ for all $\varphi$ separately*. To simplify explanations, we eliminate the skill index $\varphi$ from $\Phi$ and $LT$ unless otherwise desired for a clarification.

All learning trajectories in a given log data are consolidated into a single learning trajectory graph (LTG). An LTG is an instance of MDP. In the LTG, states represent *learning status* and actions represent instructional components taken that caused a change in learning status.

We define a *learning status* for student $i$ at time $T$ for a particular skill $\varphi$ as a pair of action history and mastery level $<\boldsymbol{ah_{i,T}}, p_{i,T}(\varphi)>$ representing an intermediate state of learning. Action history $\boldsymbol{ah_{i,T}}$ is a binary vector $<ah_i^1, \ldots, ah_i^K>$ where $ah_i^m$ shows whether student $i$ has taken the $m$-th instructional component in $\Phi^\varphi$ by time $T$ (assuming instructional components are ordered and $|\Phi^\varphi| = K$). Mastery level $p_{i,T}(\varphi)$ is a scalar value showing a predicted probability of student $i$ applying skill $\varphi$ correctly, should he/she answer an assessment quiz for the skill $\varphi$ at time $T$. The value of mastery level is rounded down to the nearest multiple of 0.05 (e.g., 0.12 becomes 0.10).

Mastery level, $p_{i,T}(\varphi)$, will be computed based on the learning trajectory with an underlying assumption that commitment to a particular type of learning activity increases mastery level by a specific amount. There are several known techniques to model this phenomenon including Bayesian-based models (e.g., Corbett and Anderson 1995) and regression models (e.g., Chi et al. 2011). As long as mastery level is monotonically increased, any student-modeling technique works for the RAFINE method.

To consolidate individual students' learning trajectories into a single learning trajectory graph (LTG), each student's learning trajectories are first converted into a *learning trajectory path*. This is done by chronologically traversing a learning trajectory while creating states each representing an intermediate learning status $<\boldsymbol{ah}_{i,T}, p_{i,T}(\varphi)>$. While traversing the learning trajectory, $\boldsymbol{ah}_{i,T}$ and $p_{i,T}(\varphi)$ are updated accordingly. For example, assume there are six instructional components: Video1, Video2, Quiz1, Quiz2, Hint1, and Hint2. A state *s* <101000, 0.4> indicates that a student has watched Video1 and taken Quiz1 before reaching the state *s*. It also indicates that a predicted mastery level at the time of arriving at the state *s* was 0.4. Assume that the student answered Quiz1 incorrectly to reach the state *s*. Now, the student wants to review Hint1, which caused a transition from *s* to *s*' where *s*' is <101010, 0.45> with an assumption that reviewing a hint increases the mastery level by 0.05.

A learning trajectory path is a linear graph. It might have loops back to the same state when a certain instructional component was taken more than once with an increase of mastery level less than 0.05. As a side note, moving between pages on the courseware is not encoded in the LTG, because it is not considered as a learning activity.

All individual students' learning trajectory paths are then aggregated into an LTG by merging the same states, which makes the LTGs a Markov Process. Note that in the LTG, student and time are abstracted. Therefore, in the following explanations, the indices *i* and *T* are omitted from a tuple representing a state such as $<\boldsymbol{ah}, p(\varphi)>$.

In an LTG, the states where the value of the mastery level, p(φ), is greater than a pre-defined threshold (which is usually 0.85) are called *terminal states*—meaning that students became proficient in applying skill φ. All actions from terminal states that appear in the original leaning trajectory data are discarded.

## Rewards

A reward value of a particular state depends on the mastery level, p(φ), both at the current and successor states. As an example, consider two students who landed on the same state *s*, but then took different learning activities. One student reached a successor state by answering an assessment quiz incorrectly hence p(φ) was not increased, whereas the other student watched a video hence p(φ) was increased.

A reward for state *s* where the student took a learning activity *a* to reach a successor state *s*' is defined as:

$$R(s,a,s') = \begin{cases} -0.14 & (ml(s) = ml(s') < 0.85) \\ -0.05 & (ml(s) < ml(s') < 0.85) \\ 0.95 & (0.85 \le ml(s')) \end{cases}$$

In the equations above, *ml*(*s*) returns the mastery level at the state *s*. A reward at the state *s* becomes the greatest when the successor state is a terminal state. Otherwise, the rewards are set to be small negative values so that the normal RL would find an optimal path to a terminal state while computing a policy. We assume that the mastery level grows monotonic, i.e., students never unlearn hence reward where *ml*(*s*) > *ml*(*s*') is undefined.

## Converse policy

Given the reward function *R* as mentioned above, a value function for state *s* under a policy $\pi$ is defined as follows, where $\boldsymbol{S}$ is a set of all states in a given LTG:

$$V^{\pi}(s) = \sum_{s' \in \boldsymbol{S}} T(s,\pi(s),s')(\mathrm{R}(s,\pi(s),s') + \gamma V^{\pi}(s'))$$

In the current implementation, the discount factor γ is arbitrarily set to be 0.9. A transition model *T*(*s*, *a*, *s*') is derived from the learning trajectory data *collected from actual students* as a probability of reaching state $s'$ when a learning activity $a$ was taken at state $s$.

In general, a policy suggests an action to be taken in a certain state to maximize the value function (Wiering and van Otterlo 2012). Under this framework, actions that are not chosen as a policy are not necessarily useless—the second best might be as effective as the best. We therefore propose to tweak the value iteration to compute the *worst* learning activities to be taken to achieve expected learning outcomes. We call this kind of policy a *converse policy*.

To compute a converse policy, the value function is updated as follows (*A*(*s*) shows a set of actions at state *s*):

$$V(s) \leftarrow \min_{a \in A(s)} \sum_{s' \in S} T(s,a,s')(R(s,a,s') + \gamma V(s'))$$

After the value function is converged, the action that *minimizes* the value function is identified as a *converse policy*:

$$\pi(s) = \operatorname*{argmin}_{a \in A(s)} \sum_{s' \in S} T(s,a,s')(R(s,a,s') + \gamma V^{\pi}(s'))$$

## Frequency Heuristic

Since the number of states in any given learning trajectory graph (LTG) is many times more than the number of available actions (i.e., instructional components), it is always the case that all instructional components are selected as a policy action at some states. Therefore, being selected as a converse policy action does not necessarily mean that the instructional component is ineffective.

We hypothesize that relatively ineffective instructional components tend to appear in a policy action set of a given converse policy more frequently than effective ones. We therefore introduce a frequency heuristic. That is, instructional components that *frequently* appear more than a predefined cut-off in a policy action set of the converse policy are included in a recommendation for refinement.

The empirical question is then "how frequent is frequent?" We examined two frequency cut-offs through an evaluation study as described in the results section.

## Evaluation Study

The primary research question for the current study is to ask if and how a holistic interpretation of a converse policy can be used to infer the usefulness of available actions.

To answer the primary research question, we break it down into three more specific questions.

RQ1: Can a converse policy correctly differentiate ineffective instructional components from effective ones?

RQ2: How robust is the converse policy as a detector for relatively ineffective instructional components against different conditions of learning data?

RQ3: How accurately does the frequency heuristic compose a recommendation?

As a proof of concept, we conducted an evaluation study for the RAFINE method. To apply RAFINE, the learning data must be structured to hold certain properties such as a presence of a skill model, which means that the instructional components on the online courseware must be tagged with a valid skill model. However, as far as we are aware, no such online courseware is available at this moment, and building RAFINE compatible online courseware requires a considerable amount of time.

Therefore, the current evaluation study used hypothetical learning trajectories made by simulating learning activities on mock online courseware. The results from the current simulation study justifies future efforts of building an RAFINE compatible online courseware.

### Simulation data

We created mock online courseware where all instructional components were tagged as either effective or ineffective. In this simulation, we assumed that there was only one skill involved in the courseware. Notice, as described above, when there were multiple skills involved, RAFINE had to be applied separately to each skill. Therefore, this assumption does not harm the generality of the study.

Learning trajectories were generated by simulating students' learning activities. For the sake of explanation, we use a phrase 'simulated students' to refer to hypothetical students in the simulation.

In the real world, the growth of mastery level depends on the learning activities actually taken and students' latent traits of learning. In the current simulation, the growth of mastery level was simulated using a logistic regression model as shown below:

$$p_{i,T} = \left\lfloor \frac{1}{1+e^{-Z_{i,T}}} \right\rfloor$$

$$Z_{i,T} = Z_{i,T-1} + \delta\big(c, e(a_{i,T-1})\big)$$

The $\lfloor x \rfloor$ operator is to round down the value $x$ to the nearest multiple of 0.05.

To answer the research question RQ2, we simulated several cases of online learning where the impact of the quality of instructional components on students' learning was controlled. Let $a_{i,T-1}$ be an instructional component that a simulated student $i$ took at time T-1. Logit $Z_{i,T}$ was increased by $\delta\big(c, e(a_{i,T-1})\big)$ that is a rectified random variable that follows a normal distribution, i.e., $\max\big(0, \mathcal{N}(\mu, \sigma^2)\big)$. The mean $\mu$ and the standard deviation $\sigma$ are determined by two discrete parameters: $c$ that represents the *contrast* in the increase of logit between effective and ineffective instructional elements—large vs. moderate vs. small, and $e(a_{i,T-1})$ that represents the *effectiveness* of the instructional element $a_{i,T-1}$—effective vs. ineffective.

The fundamental assumptions were that (1) the larger the contrast, the larger the differences of $\mu$ when effective vs. ineffective instructional components were taken, and (2) the larger the contrast, the smaller the $\sigma$ was. For example, when $c$ = "large", ($\mu$, $\sigma$) was (0.5, 0.01) vs. (-0.1, 0.01) for $e(a_{i,T-1})$ = "effective" vs. "ineffective," however they were (0.3, 0.1) vs. (0.1, 0.1) when $c$ = "small."

We also created three different versions of mock online courseware with different *qualities*. The "quality" was operationalized as a ratio of the ineffective instructional components. Each page of the mock online courseware included 3 lecture videos, 3 formative assessments, and 3 hint message each associated with an assessment. The *low*, *medium*, and *high*-quality courseware included 80-90%, 50-60%, and 10-20% ineffective instructional components.

Two instances of mock courseware were created for each level of quality. Those six instances of courseware were crossed with three levels of contrast, resulting in 18 different simulated-learning scenarios. In each scenario, simulated students took a total of 10 to 30 instructional components.

Instructional components taken were randomly determined as follows. At first, for each simulated student, the number of instructional components to be taken was randomly decided. Either a video or a quiz was then randomly selected. When it was a quiz, the correctness of the quiz response was determined randomly using the mastery level as the probability distribution. When the response was incorrect, either requesting a hint or retaking the same quiz (as the next instructional component taken) was randomly determined based on the probability distribution reported in (Aleven and Koedinger 2000). This process was repeated for the number of instructional components to be taken.

|  | **Contrast** |  |  |  |  |  |
|---|---|---|---|---|---|---|
|  | Large |  | Moderate |  | Small |  |
| **Quality** | Inef. | Ef. | Inef. | Ef. | Inef. | Ef. |
| High | 0.7±0.2 | 0.2±0.1 | 0.7+0.1 | 0.1±0.1 | 0.5±0.1 | 0.2±0.1 |
|  | (4.0) |  | (5.7) |  | (3.1) |  |
| Med. | 0.4±0.1 | 0.1±0.05 | 0.4±0.1 | 0.1±0.04 | 0.4±0.1 | 0.2±0.1 |
|  | (7.9) |  | (8.5) |  | (3.6) |  |
| Low | 0.4±0.1 | 0.04±0.04 | 0.4±0.1 | 0.04±0.03 | 0.4±0.1 | 0.1±0.1 |
|  | (9.2) |  | (10.0) |  | (4.5) |  |

Table 1: Comparison of the mean normalized frequency between ineffective (Inef.) and effective (Ef.) instructional components. A number in the parentheses shows an effect size.

Simulated students were able to retake the same instructional components.

For each of 18 learning scenarios, 100 course offerings were simulated each with 1,000 simulated students. In other words, this simulation study models a large-scale field trial as if 1800 instances of online course offerings were tested each with 1,000 student participants.

For each course-offering simulation, the learning trajectory data were converted into a learning trajectory graph (LTG). As a consequence, 1800 LTGs were created. The ad-hoc manipulation of logit described above was used to estimate mastery level in LTG (instead of actually applying a student model technique).

For each of the 1800 LTGs, the value iteration technique was applied to compute a converse policy. As a result, 1800 different converse policies were created. For each converse policy, a recommendation for refinement for a corresponding instance of online courseware was created.

## Results

To verify the feasibility of the simulation data, we computed a correlation between the ratio of effective to ineffective instructional components taken and the final mastery level. Due to a space limitation, we do not provide details about the analysis, but the data showed a very strong positive correlation, $r = 0.70$, t(1799998)=1314.56, $p < 0.001$, suggesting that the final mastery level was significantly higher when simulated students took relatively more effective than ineffective instructional components.

### Converse policy as a quality indicator

Can a converse policy correctly differentiate ineffective from effective instructional components? (RQ1) And, how robust is the converse policy as a detector for ineffective instructional components against different conditions of learning data? (RQ2) To answer these questions, we compared the frequency of individual instructional components selected as a converse policy action.

The frequency of an instructional component $\iota$ selected as a converse policy action was normalized as follows. Let $\pi$ be a converse policy and $\mathbf{S}$ be the set of states in the LTG. Let $\mathbf{S}^{\pi}(\iota) = \{s | \pi(s) = \iota\}$, i.e., a set of states in the LTG where $\iota$ is the policy action. Let $\mathbf{S}^{\mathcal{A}}(\iota) = \{s | \iota \in \mathcal{A}_s\}$ ($\mathcal{A}_s$ is a set of actions available from state $s$), i.e., a set of states where the instructional component $\iota$ was taken. Notice that $\mathbf{S}^{\pi}(\iota) \subset \mathbf{S}^{\mathcal{A}}(\iota)$. The normalized frequency of instructional component $\iota$ is the ratio $|\mathbf{S}^{\pi}(\iota)| / |\mathbf{S}^{\mathcal{A}}(\iota)|$.

We then tested if there was a significant difference in the mean normalized frequencies between effective and ineffective instructional components. Table 1 shows the mean normalized frequencies of ineffective and effective instructional components and those standard deviations. The effect size is a ratio of the difference between two means to the standard deviation. Table 1 suggests that regardless of the quality and contrast, ineffective instructional components were selected as a converse policy action notably many times more than effective ones (the differences were all statistically significant though statistics are omitted due to the space constraint). The data also suggest that the difference in the frequencies between ineffective and effective components becomes the smallest (as indicated by the smallest effect size) when contrast is small and quality is high, as we expected.

These results support the frequency heuristic hypothesis that *relatively ineffective instructional components tend to appear in a converse policy action set more frequently than effective instructional components*. It is also shown that *the converse policy is robust enough to discriminate the effectiveness of the instructional component regardless of the quality of the online courseware (operationalized as the ratio of effective vs. ineffective components) and the contrast (operationalized as the difference in the growth of logit between effective and ineffective components)*.

### Frequency heuristic for recommendation

How accurately does the frequency heuristic compose a recommendation? (RQ3) To answer this question, the relevance measure (recall and precision) for instructional components taken into a recommendation was computed.

As a reminder, those instructional components whose normalized frequency is more than a pre-defined cut-off are labeled as "ineffective" and included in the recommendation. What the cut-off value should be, however, is an empirical call.

In the current study, we tested two different cut-offs using the mean (M) and the standard deviation (SD) of the normalized frequencies of individual instructional components: M+SD and M–SD.

Figure 2 shows precision and recall scores comparing M+SD (red dashed lines) and M–SD (blue solid lines) cut-offs crossed with the quality of the courseware. For each data point, three levels of contrast are aggregated, because

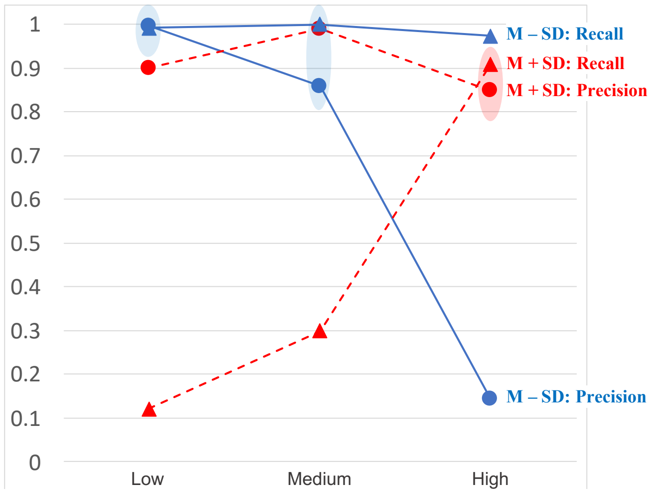


Figure 2 : Precision and recall of a recommendation. The X-axis represents the quality of the courseware. Red dashed lines show results from M+SD and blue solid lines show M–SD.

there was no notable difference among them.

The figure shows that when the quality of courseware is low to medium, the M–SD cut-off had better recall and precision scores than M+SD. F1 score for M–SD was 0.99 and 0.92 for low and medium qualities respectively. On the other hand, when the quality is high, the M+SD cut-off outperformed M–SD. F1 score of M+SD for high quality courseware was 0.88.

In sum, *the frequency heuristic adequately works to determine which instructional components must be taken as a recommendation for courseware refinement*. In the current simulation study, *over 90% of ineffective instructional components were correctly taken into a recommendation when an appropriate cut-off was used based on the maturity of the courseware*. When the courseware is newly built (which is usually in a low to medium quality), the M–SD cut-off should be used, whereas the M+SD cut-off should be used for matured (high-quality) courseware. In the currently study, even with the high-quality courseware (where only 10-20% of all instructional components on the courseware are ineffective), RAFINE was able to correctly include ineffective components in the recommendation with the M+SD cut-off.

## Discussion

The current study demonstrated that *a policy generated by reinforcement learning can be holistically interpreted to provide an insight into the usefulness of actions*. In the case of current context, the actions correspond to the instructional components used on online courseware that is subject to refinement. In particular, the results from the current study showed that when the policy represents the least optimal actions, which we call the converse policy, the holistic interpretation of the converse policy can accurately identify ineffective instructional components.

Our central hypothesis on the frequency heuristic was supported. Counting the frequency of individual instructional components appearing in a converse policy action set becomes a reliable basis for determining instructional components that have relatively little contribution to students' learning. The robustness of the identification turned out to be quite high—even when the ratio of the ineffective instructional components on a courseware was very low, RAFINE still successfully included ineffective ones in a recommendation (Figure 2).

The current RAFINE method relies on the underlying skill model. If the skill model does not adequately reflect actual latent skills, then the output from RAFINE becomes unreliable. The skill model must be validated apart from the RAFINE method. The learning factor analysis (Cen et al. 2006), in particular, may be used to improve the validity of the skill model.

One question that is not addressed in the current study is about the students' differences—how much the students' individual differences affect the "effectiveness" of each instructional component. Instructional components that are quite effective for one group of students may not be as effective for another group of students. Although it is out of the scope of the current paper, we have two working hypotheses for future studies. One hypothesis is about the majority rule—the big data overrides the individual human factors and detects the latent trends. Another hypothesis is about the individualized student model—entering individual student factors into the student model used to compute the mastery level, e.g., the individualized additive factor model (Yudelson, Koedinger and Gordon 2013). Further study will be necessary to address these issues in detail.

## Conclusion

We found that a converse policy computed via reinforcement learning applied to learning data on existing online courseware can be analytically interpreted as a whole to accurately identify relatively less effective instructional components actually used by the students. The results from a simulation study suggest a potential of the RAFINE method for evidence-based learning engineering.

As the major theoretical contribution of the current work, we demonstrated that the *holistic interpretation over the converse policy is potentially a powerful analytic tool for the quality control of practical systems*. For future studies, it is crucial to measure the actual effectiveness of the proposed method in authentic learning settings by building RAFINE compatible courseware and apply the method to real students' learning data. Though building the RAFINE compatible courseware would be quite expensive, the current paper justifies such effort.

## References

Aleven, V., and Koedinger, K. R. 2000. Limitations of student control: Do students know when they need help? In G. Gauthier, C. Frasson, & K. VanLehn (Eds.), *Proceedings of 5th International Conference on Intelligent Tutoring Systems* (pp. 292-303): Springer Verlag.

Azizsoltani, H.; Kim, Y. J.; Ausin, M. S.; Barnes, T.; and Chi, M. 2019. Unobserved Is Not Equal to Non-existent: Using Gaussian Processes to Infer Immediate Rewards Across Contexts. In S. Kraus (Ed.), *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence (IJCAI)* (pp. 1974-1980): IJCAI Organization.

Cen, H.; Koedinger, K. R.; and Junker, B. 2006. Learning Factors Analysis – A General Method for Cognitive Model Evaluation and Improvement. In M. Ikeda, K. Ashley, & T.-W. Chan (Eds.), *Intelligent Tutoring Systems* (Vol. 4053, pp. 164-175): Springer Berlin Heidelberg.

Chi, M.; Koedinger, K. R.; Gordon, G.; Jordan, P.; and VanLehn, K. 2011. Instructional factors analysis: A cognitive model for multiple instructional interventions. In J. Stamper, Z. Pardos, M. Mavrikis, & B. M. McLaren (Eds.), *Proceedings of the 4th International Conference on Educational Data Mining* (pp. 61-70).

Corbett, A. T., and Anderson, J. R. 1995. Knowledge tracing: Modeling the acquisition of procedural knowledge. *User Modeling and User Adapted Interaction, 4*(4), 253-278.

Dede, C.; Richards, J.; and Saxberg, B. (Eds.). (2018). *Learning Engineering for Online Education: Theoretical Contexts and Design-Based Examples*: Routledge.

Fishman, B.; Marx, R. W.; Blumenfeld, P.; Krajcik, J.; and Soloway, E. 2004. Creating a Framework for Research on Systemic Technology Innovations. *The Journal of the Learning Sciences, 13*(1), 43-76. doi:10.2307/1466932

Guàrdia, L.; Maina, M.; and Sangrà, A. 2013. MOOC design principles: A pedagogical approach from the learner's perspective. *elearning papers*(33).

Koedinger, K. R.; Corbett, A. T.; and Perfetti, C. 2012. The Knowledge-Learning-Instruction Framework: Bridging the Science-Practice Chasm to Enhance Robust Student Learning. *Cognitive Science, 36*, 757-798. doi:10.1111/j.1551-6709.2012.01245.x

Martin, B.; Mitrovic, A.; Koedinger, K. R.; and Mathan, S. 2011. Evaluating and improving adaptive educational systems with learning curves. *User Modeling and User-Adapted Interaction, 21*(3), 249-283. doi:10.1007/s11257-010-9084-2

Milligan, S., and Griffin, P. 2016. Understanding learning and learning designin MOOCs: A measurement-based interpretation. *Journal of Learning Analytics, 3*(2), 88-115. doi:http://dx.doi.org/10.18608/jla.2016.32.5

Rafferty, A. N.; Brunskill, E.; Griffiths, T. L.; and Shafto, P. 2015. Faster Teaching via POMDP Planning. *Cogn Sci, 40*(6), 1290-1332. doi:10.1111/cogs.12290

Shapiro, H. B.; Lee, C. H.; Wyman Roth, N. E.; Li, K.; Çetinkaya-Rundel, M.; and Canelas, D. A. 2017. Understanding the massive open online course (MOOC) student experience: An examination of attitudes, motivations, and barriers. *Computers & Education, 110*, 35-50. doi:http://dx.doi.org/10.1016/j.compedu.2017.03.003

Sutton, R. S., and Barto, A. G. 1998. *Reinforcement Learning: An Introduction*: MIT Press.

Wang, P.; Rowe, J.; Min, W.; Mott, B.; and Lester, J. 2017. Interactive narrative personalization with deep reinforcement learning. In C. Sierra (Ed.), *Proceedings of the 26th International Joint Conference on Artificial Intelligence (IJCAI)* (pp. 3852-3858). Melbourne, Australia: IJCAI Press.

Wiering, M., and van Otterlo, M. (Eds.). (2012). *Reinforcement Learning*. Heidelbereg, Berlin: Springer.

Yudelson, M.; Koedinger, K.; and Gordon, G. 2013. Individualized Bayesian Knowledge Tracing Models. In H. C. Lane, K. Yacef, J. Mostow, & P. Pavlik (Eds.), *Artificial Intelligence in Education* (Vol. 7926, pp. 171-180): Springer Berlin Heidelberg.

Zhang, J.; Hao, B.; Chen, B.; Li, C.; Chen, H.; and Sun, J. 2019. Hierarchical Reinforcement Learning for Course Recommendation in MOOCs. In *Proceedings of the AAAI Conference on Artificial Intelligence* (pp. 435-442): AAAI Press.