KCluster: An LLM-based Clustering Approach to Knowledge Component Discovery

Yumou Wei
Human-Computer Interaction Institute
Carnegie Mellon University
yumouw@cs.cmu.edu

Paulo Carvalho
Human-Computer Interaction Institute
Carnegie Mellon University
pcarvalh@cs.cmu.edu

John Stamper
Human-Computer Interaction Institute
Carnegie Mellon University
jstamper@cs.cmu.edu

ABSTRACT
Educators evaluate student knowledge using knowledge component (KC) models that map assessment questions to KCs. Still, designing KC models for large question banks remains an insurmountable challenge for instructors who need to analyze each question by hand. The growing use of Generative AI in education is expected only to aggravate this chronic deficiency of expert-designed KC models, as course engineers designing KCs struggle to keep up with the pace at which questions are generated. In this work, we propose KCluster, a novel KC discovery algorithm based on identifying clusters of congruent questions according to a new similarity metric induced by a large language model (LLM). We demonstrate in three datasets that an LLM can create an effective metric of question similarity, which a clustering algorithm can use to create KC models from questions with minimal human effort. Combining the strengths of LLM and clustering, KCluster generates descriptive KC labels and discovers KC models that predict student performance better than the best expert-designed models available. In anticipation of future work, we illustrate how KCluster can reveal insights into difficult KCs and suggest improvements to instruction.

Keywords
Knowledge Component, Large Language Model, Clustering

1. INTRODUCTION
Real knowledge is to know the extent of one's ignorance—as Confucius reflected on his epistemology. One way educators can evaluate student knowledge, according to the Knowledge-Learning-Instruction (KLI) framework [23], is by developing cognitive models that map assessment items (or questions) to knowledge components. A **knowledge component** (KC) is a unit of cognitive function or structure that a student acquires through learning [23], representing specific information, concepts, or skills that a student needs to solve a task or a problem—a student must know how to "use guide words" before determining whether "guess" can be found on a dictionary page marked with "garage" and "goose". With a well-designed cognitive model (or **KC model**), instructors can divide a complex topic into simpler and more manageable milestones that help track student learning [9], identify learning sub-goals with which students struggle [46], and organize instruction events to promote knowledge transfer [28].

Despite the numerous benefits of a well-designed KC model, mapping assessment questions to KCs still remains an insurmountable challenge for instructors and instructional designers who are overwhelmed by the sheer amount of questions that each need to be analyzed by hand. Cognitive Task Analysis (CTA) [7], the *de facto* best manual approach to KC discovery, incurs considerable labor and time costs that prevent schools and teachers from gaining equitable access; therefore, many datasets that naturally occur from students interacting with educational technologies lack **expert-designed KC models**. For example, nearly 60% of the 4,639 datasets available in DataShop [47]—the largest educational data repository—do not contain more significant KC models than the default `Single-KC` and `Unique-step` models that are only intended to serve as benchmarks[1]. This absence of expert-designed KCs limits the analytics that can be conducted and the educational insights that such data can provide. Furthermore, we expect that the increasing adoption of Generative AI (GenAI) in education can only exacerbate this deficiency, as learning engineers developing KCs struggle to keep up with the pace at which questions are produced by GenAI and become even less likely to provide quality KCs.

This chronic deficiency of expert-designed KCs in large question banks, aggravated by the accelerating use of GenAI in education, calls for a new effective KC discovery algorithm that can automatically extract KCs from abundant question content with minimal burden on instructors. A notable approach, SMART [35], extracts KCs from instructional content based on the assumption that a cluster of linguistically similar texts shares the same KC. SMART applies k-means **clustering** to the TF-IDF embeddings of instructional texts and obtains descriptive KC labels using a keyword extraction algorithm called TextRank [36]. Although shown in two science datasets to create KC models that predict student responses better than expert-designed models do, SMART still requires a course engineer to specify the number of KCs to discover—a hyperparameter that the authors reported has a statistically significant impact on how well SMART

[1]Based on DataShop administrators' response to our inquiry in December 2024

Yumou Wei, Paulo Carvalho, and John Stamper. KCluster: An LLM-based Clustering Approach to Knowledge Component Discovery. In Caitlin Mills, Giora Alexandron, Davide Taibi, Giosuè Lo Bosco, and Luc Paquette (eds.) Proceedings of the 18th International Conference on Educational Data Mining, Palermo, Italy, July, 2025, pp. 228–240. International Educational Data Mining Society (2025).

https://doi.org/10.5281/zenodo.15870197

fits to the student data; moreover, identifying each KC with short keywords, SMART tends to produce coarse labels that result in *identical* labels for what experts believe should be *separate* KCs. A more recent approach [37] uses a **large language model** (LLM) to identify KCs for multiple-choice questions. The authors implemented two strategies—simulated expert and simulated textbook—that encourage the LLM to generate descriptive KC labels based on question content. Although in an evaluation study involving three participants, the majority preferred the LLM-generated KC labels to those crafted by experts for more than 60% of the evaluated questions, this LLM-based approach, contrary to SMART, produces *slightly different* labels for questions that experts believe should belong to the *same* KC, as acknowledged by the authors. The two current divergent approaches to KC discovery beg the question: **Will a hybrid of clustering and LLM produce synergy in extracting KCs from questions?**

In this work, we propose `KCluster`[2], an unsupervised **KC** discovery algorithm based on identifying **cluster**s of *congruent* questions according to a novel similarity metric induced by an LLM. By extending word collocations to questions, we developed a novel concept called **question congruity** that quantifies the similarity of two questions by the likelihood of their co-occurrence, and devised an algorithm that uses LLM as a probability machine to compute the required text probabilities **without retraining or finetuning the LLM**. Combining the strengths of LLM and clustering, `KCluster` uses *Phi-2* [20] (an LLM) to measure question congruity and generate descriptive KC labels, and uses *affinity propagation* [14] (a clustering algorithm) to identify clusters of congruent questions, each corresponding to a KC. We validated `KCluster` on three datasets related to science and e-learning, two of which contain student response data, giving affirmative answers to our three research questions (RQs):

- RQ-1: Does `KCluster` align with expert-designed KC models? (Section 5.1)
- RQ-2: Does `KCluster` enable accurate prediction of student responses? (Section 5.2)
- RQ-3: Does `KCluster` reveal insights about problematic KCs? (Section 5.3)

Through our comprehensive evaluation comparing `KCluster` to three other competitive methods on large question banks and student data, we demonstrate that **an LLM can create a new, effective measure of similarity between two arbitrary questions, which a clustering algorithm can use to extract KCs from questions automatically, without elaborate retraining, finetuning, or prompt engineering**. The main contributions of our research include: 1) a novel measure of question similarity, 2) an algorithm to compute the new similarity metric using LLM, and 3) an effective approach to extract descriptive KC labels from question content.

2. LITERATURE REVIEW

A comprehensive review of the literature on KC discovery is necessary to show how `KCluster` connects to and builds on current approaches. We classify the approaches into three categories based on the amount of manual work required and review them in decreasing order of human involvement.

2.1 Manual Approaches

Manual approaches rely solely on the expertise of an instructional designer to identify KCs. Although a teacher could review and label each question with a KC, a more systematic approach is through Cognitive Task Analysis (CTA) [7], where instructional experts are asked to elucidate their mental processes in solving problems during a think-aloud interview. A notable CTA approach is Difficulty Factors Assessment (DFA) [19, 26], based on the assumption that students should perform similarly on questions concerning the same KC—therefore, any performance discrepancy is due to a hidden KC yet to be discovered. For example, using DFA, researchers identified a new KC (about comprehending the symbolic representation of quantitative relations) that explained why beginning algebra students performed worse on algebra problems presented with mathematical symbols than on problems embedded in a hypothetical story, illuminating the effect of problem presentation on learning that had been overlooked [26]. Although CTA is known to improve instruction [24], the outcome is highly sensitive to the CTA methods used and the instructional context considered [49]. Moreover, CTA relies heavily on experts to make subjective decisions and therefore incurs considerable labor and time costs that prevent CTA from scaling to large question banks readily available with GenAI. (Semi-)automated approaches, however, alleviate the scalability problem by minimizing human involvement and learning KC models from data.

2.2 Semi-automated Approaches

Semi-automated approaches refine an expert-designed KC model with data-driven methods. A notable approach [46] extends DFA with a statistical model of student data to identify problematic KCs worth improving; by analyzing a difficult KC identified from data, researchers uncovered three hidden KCs for geometry area learning and obtained a better prediction of student performance. In a sequel [27], researchers reaffirmed the efficacy of this data-driven DFA approach by redesigning a cognitive tutor for teaching geometry and showing improvements in student learning. An alternate approach, Learning Factors Analysis (LFA) [2], further automates DFA by using the A^* algorithm [43] to search for better KC models based on a list of difficult factors that experts think are absent from the current model. In an evaluation study [25] researchers found LFA improve KC models across ten datasets of various domains and closed the development-test-redesign loop in a sequel [33] that redesigned a tutoring system using LFA-generated insights. Although semi-automated approaches are grounded on student data, they rely on expert-designed KC models to produce descriptive KC labels, calling for more automated approaches that eliminate human input.

2.3 Automated Approaches

Automated approaches develop new KC models from scratch and do not require human input beyond a few hyperparameters. The Q-matrix method [1] and its sequels using matrix factorization [11, 12, 29] search for a KC model that best predicts student responses to questions. A closely related

[2]Pronounced the same as "cluster", `KCluster` is freely available at `https://github.com/weiyumou/KCluster`.

class of approaches discovers KCs as part of a statistical model learned from data—one method [32] creates KC models through a DINA model [10], while dAFM [39] and SparFAE [38], both using neural networks, estimate Q-matrices via an AFM [3] and an IRT model [18]; other similar approaches have explored Hidden Markov Model [16] and extended to identifying KCs in programming problems [45]. These automated approaches based on statistical learning, although capable of identifying KCs without human intervention, still require reference to an expert-designed KC model to produce descriptive KC labels (otherwise, they produce nominal labels such as "KC-15", which provides no instructional insights); therefore, they are better suited for unsupervised *KC refinement* than automatic KC discovery.

A unique class of automated approaches that can produce descriptive KC labels without a reference model extracts KCs from *instructional content* such as textbooks. For example, SimStudent-based approach [30, 31] iteratively associates predefined skill labels with *problem-solving demonstrations* and creates new KC labels if necessary; similarly, researchers have explored a term-matching approach to extract concepts from *student explanations* for math problems [44]. Another approach, FACE [4], identifies concepts from *adaptive textbooks* based on an extensive list of hand-engineered features. All these approaches, however, require a list of key skills or concepts specified by experts beforehand. A notable approach that does not require human input, SMART [35], extracts KCs from instructional texts and questions by clustering similar texts encoded as TF-IDF vectors. The k-means clustering algorithm was applied to both the embedding vectors and their cosine similarity, although no significant differences were observed; the researchers then applied TextRank [36] to extract keywords from each of the k clusters to use as KCs. Although SMART was validated on two science datasets to create quality KC models, it still required an expert to specify k, the number of KCs to discover, and the keywords identified by TextRank were so coarse that resulted in duplicate labels for what experts believe should be distinct KCs. A more recent approach [37] uses an LLM to identify KCs from multiple-choice questions by asking the LLM to simulate instructional experts or textbook authors. Although in a three-subject evaluation study, the majority of the evaluators showed preference for LLM-generated KC labels in more than 60% of the evaluated questions, this approach produces an excessive number of KCs because, in contrast to TextRank, LLM is so capable that it generates slightly different labels for questions that experts believe should belong to the same KC. The two divergent approaches suggest that clustering, capable of uncovering latent question structures, and LLM, capable of generating descriptive KC labels, can form synergy in KC discovery.

3. METHODS

We propose, evaluate, and compare three classes of automated KC extraction methods, each of which extends the preceding method and builds upon a large language model (LLM). The LLM we used in this work is Phi-2 [20] from Microsoft, a lightweight open-source model trained on high-quality textbook-like data [17] and potentially suited for educational data mining. We used the Phi-2 distribution freely available through HuggingFace [52] and used PyTorch [40] for our custom implementation. Phi-2 was deployed to a computing cluster with access to NVIDIA A40 GPUs.

Table 1: The prompt template used in `Concept` (left) and a concrete prompt filled with an actual question (right).

`Exercise 1:`	`Exercise 1:`
`{question type}:`	`Multiple Choice:`
`{stem}`	`Which is the most flexible?`
`{`	`a) paper`
`    choices`	`b) ceramic tea cup`
`}`	`c) clay tile`
`Answer: {answer}`	`Answer: a) paper`
`Remark:`	`Remark:`
`The above exercise is a`	`The above exercise is a`
`{question type} question`	`multiple-choice question`
`that tests whether the`	`that tests whether the`
`student understands the`	`student understands the`
`concept of [extracted KC]`	`concept of [extracted KC]`

Although having fewer parameters (2.7B) than most mainstream models, Phi-2 is ideal for building our KC discovery algorithms and providing resource-constrained institutions with equitable access to GenAI tools because it offers a good balance between performance and affordability [20]. Our choice of Phi-2 may seem unconventional, when "GPT" has nearly become synonymous with LLMs. However, we believe that Phi-2 offers two distinct advantages that make it a compelling choice for our research. First, Phi-2 is an open-source model, which allows us to access its hidden states and output log-probabilities essential for developing our KC extraction methods; as shown in Section 3.3, `KCluster` requires us to evaluate the log-probability of *any* token, whereas the OpenAI API[3] only supports the top 20 most likely tokens *that it returns*. Second, under modest hardware requirements, Phi-2 is overall the best LLM with <10B parameters, outperforming Mistral 7B [21] and Llama-2 13B [50] in math [8] and coding [5] tasks; smaller or earlier models like BERT [13] would not have benefited from the extensive pre-training on large textbook-like corpora that made Phi-2 potentially suitable for educational tasks. Building our three KC extraction methods with Phi-2 represents a leading effort to explore the potential of alternative LLMs for educational applications, such as KC discovery.

3.1 Concept Extraction

A straightforward application of LLMs to KC discovery is to extract concepts from questions. In line with previous work using LLMs [37], we explicitly ask Phi-2 to identify the *key concept* that a student must know to answer a question correctly and treat each concept as a KC. Through extensive prompt engineering, we discovered an effective *prompt template*, which allowed us to obtain descriptive and accurate concept labels without elaborate prompting strategies as used in previous work [37]. Shown on the left of Table 1, the prompt template includes special markers to which Phi-2 is particularly responsive. For example, we discovered that the marker "`Exercise 1:`" followed by `{question type}` prompts Phi-2 to generate a new question in a format that we now adopt in the prompt template (namely,

[3] `https://platform.openai.com/docs/api-reference/chat/create#chat-create-top_logprobs`

Table 2: Code for obtaining question embeddings from Phi-2; speacial markers like `Exercise 1` are passed to the `text` parameter of the tokenizer and question texts to `text_pair`. Only embeddings of the question texts are obtained.

```
inputs = tokenizer(text=["Exercise 1:..."], text_pair=["Which is the..."],
                   return_tensors="pt", return_token_type_ids=True, padding=True).to(device)
mask = inputs.pop("token_type_ids").bool() # useful for distinguishing markers from question content
indices = torch.cumsum(mask.sum(-1), dim=-1).tolist() # the starting index of each question's text
last_states = model(**inputs, output_hidden_states=True).hidden_states[-1] # a loaded Phi-2 model
chunks = torch.vsplit(last_states[mask], indices)[:-1]
embeddings = torch.stack([torch.mean(c, dim=0) for c in chunks], dim=0)
```

`stem`, `choices`, and `Answer:`). Similarly, "`Remark:`" encourages Phi-2 to write a comment starting with "The above exercise..." about the preceding question; therefore, we expanded the remark with more explicit instructions asking Phi-2 to complete generation with the key concept. None of these special markers are officially documented [20], but are discovered from our extensive prompt engineering. On the right of Table 1 shows a concrete prompt derived from the template by replacing the variables in curly brackets with specific values. We denote this method as `Concept`.

In generating the key concepts, we adopt a *greedy decoding strategy*, in which Phi-2 always selects the most probable token at each generation step. Moreover, we use beam search to maintain five candidate concepts during generation and apply a length penalty [53] to encourage Phi-2 to generate succinct concepts—for the example prompt shown in Table 1, Phi-2 produced "flexibility". Generation stops when a period or comma appears, and we select the best candidate with the highest probability. As shown in Section 5, using concepts as KCs, `Concept` is a competitive baseline that produces KC labels in reasonable alignment with expert-crafted ones; it is also used by other KC discovery algorithms described hereinafter to create descriptive KC labels.

3.2 Semantic Embedding

A known limitation of `Concept`, as encountered in previous work [37], is that the LLM can generate *slightly different* KC labels for questions to which an instructional expert would assign the *same* KC—the single and plural forms of the same concept (gas vs. gases), among other trivialities, can result in redundant labels that could have been merged. One approach to reducing such redundancy, as used in SMART [35], is to group similar instructional items by applying a clustering algorithm to their *semantic embeddings* and assign each group to a KC. Depending on which item we convert to embeddings, we introduce two embedding-based methods as enhanced baselines.

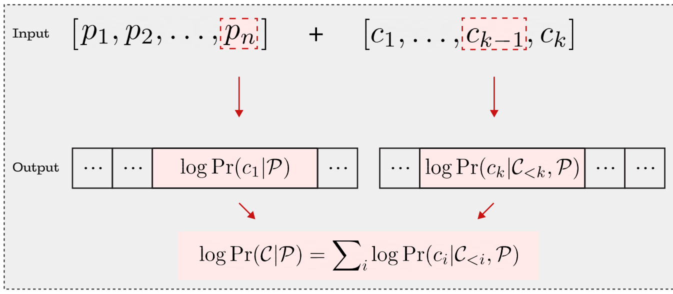
- **Concept embedding**: A natural extension to `Concept` is to encode the key concepts extracted by Phi-2 as vectors and assign questions to KCs based on *concept similarity*. Since each concept is a short phrase, we use a state-of-the-art sentence embedding model, Sentence Transformer [41] with "`all-mpnet-base-v2`" backend that offers the best quality, to produce a vector of 768 dimensions for each concept. We call this method `Concept-emb`.
- **Question embedding**: An alternative is to encode the questions, which contain more information than the concepts, and group the questions based on *question similarity*. We present questions to Phi-2 using the same prompt template shown on the left of Table 1 (without `Remark`), and take the 2560-dimensional average vector of Phi-2's last hidden states before the language-modeling head as question embeddings (a code snippet is listed in Table 2 for reference). We call this method `Question-emb`.

Since the two methods produce embeddings of different sizes, to ensure a fair comparison, we further reduce the embeddings to their similarities. As shown in SMART [35], using similarity rather than embeddings does not affect the quality of the resulting KC models—if not more advantageous. In particular, we use *negative cosine distance*[4], defined as $\cos(\mathbf{x}, \mathbf{y}) - 1$ for two vectors $\mathbf{x}$ and $\mathbf{y}$, to quantify the similarity between two embeddings (the values range from -2 to 0, with identical vectors having the largest value of 0).

After we obtain the similarity matrix of the embeddings, we use clustering to identify questions that share similar concepts (as in `Concept-emb`) or content (as in `Question-emb`). The clustering algorithm we used is *affinity propagation* [15], which does not require the number or the size of the clusters to be pre-specified; instead, it takes as input a matrix describing the affinity between input items and discovers item clusters through optimization. Each cluster is uniquely identified by its central item called "exemplar", and the user can specify an initial *preference* for each input item to be an exemplar. The algorithm is so named because it propagates between items two kinds of *messages* derived from the affinity matrix: at every iteration, an item i sends to another item j a number (the message) reflecting the *responsibility* for i to choose j as an exemplar over others, and receives from j another number indicating j's *availability* to be an exemplar of i with respect to other items that have chosen j as an exemplar. In essence, affinity propagation stimulates the items to compete for being an exemplar and halts when the exemplars (and the clusters) stop changing. In addition to not requiring the number of clusters be specified, affinity propagation accepts affinity measures that are not necessarily a mathematical metric, allowing the use of task-specific measures that are expected to result in better performance.

We set a uniform preference (using the median affinity of all pairs of input, by default) for each concept or question to be an exemplar. At convergence, affinity propagation produces a nominal cluster label for each input item and a one-to-one mapping of exemplars to clusters. While questions within a

[4] It is equivalent to cosine similarity but more compatible with other negative distances that future work may explore.




```
inputs = tokenizer(text=P, text_pair=C,
          return_tensors="pt", return_token_type_ids=True,
          padding=True).to(device)
mask = torch.logical_not(inputs.pop("token_type_ids"))
labels = torch.masked_fill(inputs["input_ids"], mask, -100)
```

```
logits = model(**inputs).logits.transpose(-1, -2)
```

```
loss = cross_entropy(logits[..., :-1], labels[:, 1:],
                     reduction="none", ignore_index=-100)
log_probs = -torch.sum(loss, dim=-1)
```

Figure 1: An illustration of the algorithm for computing $\log \Pr(\mathcal{C}|\mathcal{P})$, along with three code snippets for each key step

cluster are assigned the same KC, for both `Concept-emb` and `Question-emb`, we label each question with the concept of its exemplar that we obtained from `Concept`. In practice, we always run `Concept` for all questions before running either embedding-based method to ensure that every cluster has a descriptive label, whichever questions become exemplars. If two exemplars have identical concepts, two previously separate KCs may be (unintentionally) merged, but practitioners can always choose whether or not to merge those KCs, depending on whichever leads to better performance. As shown in Section 5, using a classic similarity measure (negative cosine distance), `Question-emb` significantly outperforms `Concept` and produces less redundant KC labels.

3.3 KCluster

Using the classic cosine-based metric to measure concept or question similarity misses an opportunity to fully exploit an LLM's capability—after all, producing question embeddings is perhaps not the best use of an LLM. In addition to generating text as in `Concept`, a large *language model* is also an exceptional "probability machine" that can evaluate the probability of an arbitrary piece of text [22], **even without retraining or finetuning**. Our main KC extraction method, `KCluster`, retains the use of affinity propagation to group similar questions, but extends `Question-emb` with a new measure of question similarity based on text probabilities. We introduce *question congruity*, a new similarity metric derived from quantifying the likelihood of question collocations, and describe an algorithm that uses Phi-2 to compute the required probabilities.

3.3.1 Collocating questions are congruent

In a coherent speech, words are not uttered haphazardly but join other *congruent* words to form *collocations* (e.g., "data mining"); therefore, if one word makes the other more likely to appear in a sentence than otherwise, the two words are congruent. Since questions are made up of words, the notion of congruity can be extended from words to questions. Based on instructional design principles, we postulate that, as two words collocate in a sentence to form a phrase, **two questions can co-occur (in a worksheet or an exam paper) if they belong to the same unit, the same lesson, or better still, *the same KC***. To quantify the collocation of two questions, q_s and q_t, we consider how much *more likely* the presence of q_t makes q_s to appear, by evaluating the change in log-probabilities of q_s with and without q_t, and defining:

$$\Delta(q_s, q_t) := \log \Pr(q_s|q_t) - \log \Pr(q_s) \tag{1}$$

The underlying principle is similar to that for words: if one question significantly increases the other question's likelihood of occurrence, in which case $\Delta(q_s, q_t)$ is large positive, the two questions must be highly congruent—they may use a similar language, concern a single topic, or come from the same textbook chapter, all alluding to **a shared KC**; on the other hand, if $\Delta(q_s, q_t)$ is close to zero or negative, in which case the presence of q_t does not improve (or even hurts) the likelihood of q_s appearing, the two questions are hardly congruent—they are unlikely to belong to the same KC, since they do not even co-occur often.

Equation 1 only partially quantifies question congruity as it assumes that q_t precedes q_s; however, two questions can also co-occur (and be congruent) when, conversely, q_s precedes q_t. Therefore, we take a step further to **define** question congruity formally as a symmetric quantity that equally weighs both cases of question collocation:

$$\texttt{Congruity}(q_s, q_t) := \frac{1}{2}\left[\Delta(q_s, q_t) + \Delta(q_t, q_s)\right] \tag{2}$$

Although we have independently derived it from analyzing question collocations, our notion of question congruity coincides neatly with the established concept of *pointwise mutual information* (PMI) between words [6], which has an equivalent mathematical form; by developing `KCluster`, we extend PMI to questions, which are more intricate than words.

3.3.2 LLMs are exceptional probability machines

Computing the PMI between words requires counting collocations; counting is, however, infeasible for calculating question congruity as two questions rarely, if at all, co-occur more than once in a collection of questions (e.g., a question almost never repeats itself in a well-designed exam). Instead, given a novel question pair, we need to extrapolate their collocation probabilities (in the form of $\log \Pr(q_s|q_t)$ and $\log \Pr(q_s)$) from existing data. LLMs, trained on massive corpora of diverse genres, are perfect for implementing question congruity because of their native ability to evaluate sophisticated text probabilities [22]. In this section, we describe an algorithm that uses Phi-2 to compute question congruity.

As an LLM, not only can Phi-2 extend a prompt (as in Section 3.1), but it can also evaluate the probability of alternative continuations to a given prompt. Let $\mathcal{P} := [p_1, p_2, \ldots, p_n]$ denote a prompt comprising n tokens $(p_1, \ldots, p_n)$ and $\mathcal{C} := [c_1, c_2, \ldots, c_k]$ denote a *prompt continuation* comprising k tokens $(c_1, \ldots, c_k)$. To compute log-probabilities of the form

Table 3: The prompt template used to evaluate $\log\Pr(q_s|q_t)$ with a concrete example on the right. The template begins with q_t (above the dashed line) as the conditioning prompt $\mathcal{P}$ and ends with q_s (below the dashed line) as the prompt continuation $\mathcal{C}$, whose probability is to be evaluated.

Exercise 1: {question-type-1}: {stem-1} { choices-1 } Answer: {answer-1} Exercise 2:	Exercise 1: Multiple Choice: Which is the most flexible? a) bone b) glass jar c) rubber band Answer: c) rubber band Exercise 2:
{question-type-2}: {stem-2} { choices-2 } Answer: {answer-2}	Multiple Choice: Which is the most flexible? a) paper b) ceramic tea cup c) clay tile Answer: a) paper

$\log\Pr(q_s|q_t)$, we consider q_t as the prompt $\mathcal{P}$ and q_s as a prompt continuation $\mathcal{C}$ to $\mathcal{P}$, and evaluate $\log\Pr(\mathcal{C}|\mathcal{P})$, the log-probability that $\mathcal{C}$ continues $\mathcal{P}$. The main algorithm is illustrated in Figure 1, along with three code snippets for executing each key step.

The input to Phi-2 is a concatenation of the prompt and the continuation, $\mathcal{P}+\mathcal{C}$, producing an output of the same length. At each output location, Phi-2 generates a vector whose entries after a `log-softmax` normalization are the log-probability that each token in the vocabulary is to become the output token at that location, given the input tokens that Phi-2 has seen so far—in particular, one entry in the vector corresponds to the *next token* in the input that Phi-2 has not consumed. For example, Figure 1 shows that the output vector corresponding to the *last token* p_n in $\mathcal{P}$ contains an entry equal to $\log\Pr(c_1|\mathcal{P})$, the log-probability of the *first token* c_1 in the continuation $\mathcal{C}$ conditioned on the entire prompt $\mathcal{P}$ that has been consumed before c_1 is; similarly, the output vector corresponding to the *penultimate token* c_{k-1} in $\mathcal{C}$ contains an entry equal to $\log\Pr(c_k|\mathcal{C}_{<k},\mathcal{P})$, the log-probability of the *last token* c_k in $\mathcal{C}$ conditioned on $\mathcal{P}$ and the partial continuation $\mathcal{C}_{<k}$ up to the k-th token. By the chain rule of probability, the target quantity $\log\Pr(\mathcal{C}|\mathcal{P})$ is simply the sum of these "next-token" log-probabilities starting from $\log\Pr(c_1|\mathcal{P})$, as shown at the bottom left of Figure 1, and can be calculated analogously to language modeling with a masked `cross-entropy` loss.

To construct the input $\mathcal{P}+\mathcal{C}$ to evaluate $\log\Pr(q_s|q_t)$ for two questions q_s and q_t, we use the prompt template shown on the left of Table 3. The template consists of two parts, separated by a dashed line. The upper part represents $\mathcal{P}$ in the algorithm, and sequentially contains the special marker "`Exercise 1:`" for introducing q_t, the content of q_t, and another special marker "`Exercise 2:`" for introducing q_s. The content of q_s, however, is contained in the lower part of the template, representing $\mathcal{C}$ in the algorithm. This design ensures that Phi-2 only evaluates the log-probability of q_s, while maintaining q_t as the context.

Table 4: The prompt template used to evaluate $\log\Pr(q_s)$ with a concrete example on the right. The template uses the special marker "`Exercise 2:`" as the conditioning prompt $\mathcal{P}$ and the content of question q_s (below the dashed line) as the prompt continuation $\mathcal{C}$, whose probability is to be evaluated.

Exercise 2:	Exercise 2:
{question-type-2}: {stem-2} { choices-2 } Answer: {answer-2}	Multiple Choice: Which is the most flexible? a) paper b) ceramic tea cup c) clay tile Answer: a) paper

Calculating question congruity also requires computing marginal log-probabilities of the form $\log\Pr(q_s)$, for which we use the same algorithm for computing $\log\Pr(\mathcal{C}|\mathcal{P})$ but keep $\mathcal{P}$ minimal. Table 4 shows the prompt template for computing the marginals, with a concrete example on the right. Compared to the prompt template in Table 3, the new template removes all traces of the conditioning question q_t from the upper part representing $\mathcal{P}$, but retains the special marker "`Exercise 2:`" for introducing q_s in the lower part representing $\mathcal{C}$. This design ensures the algorithm closely approximates the genuine marginal log-probability $\log\Pr(q_s)$ while keeping $\log\Pr(q_s)$ compatible with $\log\Pr(q_s|q_t)$ by only removing information about q_t.

Defined in terms of differences ($\Delta(q_s,q_t)$ and $\Delta(q_t,q_s)$), question congruity is invariant to the length of the questions, as the effect of length in $\log\Pr(q_s|q_t)$ offsets that in $\log\Pr(q_s)$, making it a versatile measure for different types and lengths of questions. Furthermore, question congruity captures more than text similarity, but an abstract notion of congruence (one question following another) that cosine-based metrics do not convey. We show in Section 5 that question congruity is more effective than negative cosine distance in measuring similar questions for clustering-based KC discovery.

4. DATASETS

We evaluate the four KC extraction methods described so far (`Concept`, `Concept-emb`, `Question-emb`, and `KCluster`) on three datasets of multiple-choice questions (MCQs) that vary in size and domain. All datasets include at least one expert-designed KC model that we consider as the gold standard in our evaluation, and two datasets contain additional data that allow us to validate each model on student transactions recorded in an actual class.

4.1 ScienceQA

Based on various grade-school science curricula, **ScienceQA** [34] is a multi-modal dataset that covers three subjects: social science, language science, and natural science. Each question has two to four choices with one correct answer and comes with a "skill" tag—such as "identify the experimental question"—that we consider as a KC label designed by an expert. To prepare the dataset for evaluation, we discarded questions accompanied by an image or tagged with a skill that appears less than ten times in all text-based questions, creating an evaluation subset of 10,701 MCQs.

Table 5: KC alignment with Skill (99 KCs) assessed on ScienceQA

		Adj. Rand [-0.5, 1]	Adj. MI $(-\infty, 1]$	FM Index [0, 1]	Homogeneity [0, 1]	Completeness [0, 1]	V-measure [0, 1]
`Concept`	(549 KCs)	0.6454	0.8177	0.6603	0.9135	0.7925	0.8487
`Question-emb`	(188 KCs)	**0.6940**	0.8437	**0.7036**	0.9001	0.8308	0.8641
`KCluster`	(198 KCs)	0.6617	**0.8513**	0.6759	**0.9157**	**0.8310**	**0.8713**

4.2 E-learning 2022

Publicly available in DataShop [47], the **E-learning 2022** data set[5] contains questions and student activity data collected in a graduate e-learning design course taught between August and December 2022—a small subset of 80 MCQs were used in previous KC extraction work [37]. We parsed the course content in HTML documents and extracted 630 MCQs corresponding to 42,176 problem-solving attempts made by 39 students. In addition to the two default KC models, `Single-KC`, where all steps are labeled with a single KC, and `Unique-step`, where each step is labeled with a unique KC, this dataset includes two expert-designed KC models based on learning objectives (LOs): `LOs` and its improved version, `LOs-new`. In contrast to previous work [37], we did not attempt to balance the number of MCQs per KC by curating a special subset of the MCQs, but retained the original mapping of MCQs to KCs in the expert-designed KC models for a more faithful evaluation of all methods.

4.3 E-learning 2023

The **E-learning 2023** dataset[6] is derived from the same e-learning course taught by a different instructor in a different semester (from August 2023 to December 2023). Unlike E-learning 2022, there was no course content available to extract questions from, so we chose 497 MCQs that are present in both years as the evaluation subset, which corresponds to 44,065 problem-solving attempts made by 41 students. This dataset also includes two expert-designed KC models: `v1-prompt-CTAmultimedia` (abbreviated as `v1-CTA`) and `v2-combined`, in addition to the two default KC models.

5. RESULTS AND DISCUSSION

We use data to evaluate `KCluster` against three competing methods and answer our three RQs introduced earlier.

5.1 Does KCluster align with expert-designed KC models? (RQ-1)

Although one can argue that no instructional expert could develop a flawless KC model and that expert opinions could diverge, alignment with expert-designed KC models provides quality assurance for automated KC extraction methods, as better alignment with human labels indicates more potential to be useful. In line with previous work [35], we quantify the alignment of two KC models by comparing how they assign questions to KCs rather than counting text matches in KC labels—therefore, two models are perfectly aligned if both group the questions the same way, even if every group has a different label. Allowing different labels for the same KC reflects the multiple ways in which different experts can describe a KC and accounts for the nuances in different labeling approaches. Since a KC label indicates group membership analogously to a cluster label, *regardless of whether a clustering algorithm is used*, we use standard metrics for *clustering performance*[7] to assess how better `KCluster` aligns with expert-designed KC models than the other methods.

The following three metrics emphasize *label agreement*: how well the predicted labels agree with the ground-truth classes. All methods are adjusted for chance, so that a random cluster assignment results in a score close to 0, whereas a perfect agreement has a score of 1:

- **Adjusted Rand Index** (Adj. Rand) [48]: a count-based measure popular in the literature;
- **Adjusted Mutual Information** (Adj. MI) [51]: an information-theoretic measure adjusted for chance;
- **Fowlkes-Mallows Index** (FM Index) [14]: a measure based on pairwise precision and recall.

The following three metrics highlight *cluster quality*: how well each predicted cluster corresponds to the original classes. Low-quality assignments have a score close to 0 and perfect clusters have a score of 1, although a random assignment with a large number of clusters can have a specious, non-zero score (these three metrics are not adjusted for chance).

- **Homogeneity** [42]: a cluster assignment is homogeneous if every cluster contains only elements from the same ground-truth class;
- **Completeness** [42]: a cluster assignment is complete if elements of the same ground-truth class are always assigned to the same cluster;
- **V-measure** [42]: the harmonic mean of homogeneity and completeness that balances both measures.

Because no study has shown that one metric is more decisive than the others in assessing the alignment of KC models, we report all six metrics to give a more faithful evaluation of the four KC extraction methods. For all metrics, we use expert-designed KC labels as the gold standard, and if there is more than one expert-designed KC model, we choose the one that best fits student data as described in Section 5.2. As no significant randomness is involved, we report the result of one execution of each method.

[5] `https://pslcdatashop.web.cmu.edu/DatasetInfo?datasetId=5426`

[6] `https://pslcdatashop.web.cmu.edu/DatasetInfo?datasetId=5843`

[7] `https://scikit-learn.org/stable/modules/clustering.html#clustering-performance-evaluation`

Table 6: KC alignment with `LOs-new` (101 KCs) assessed on E-learning 2022

		Adj. Rand [-0.5, 1]	Adj. MI $(-\infty, 1]$	FM Index [0, 1]	Homogeneity [0, 1]	Completeness [0, 1]	V-measure [0, 1]
`Concept`	(371 KCs)	0.2815	0.6337	0.3450	**0.9328**	0.7188	**0.8119**
`Concept-emb`	(101 KCs)	0.3090	0.6019	0.3220	0.7350	0.7240	0.7295
`Question-emb`	(91 KCs)	0.3533	0.6188	0.3668	0.7218	0.7439	0.7326
`KCluster`	(114 KCs)	**0.4553**	**0.6939**	**0.4680**	0.8139	**0.7807**	0.7970

Table 7: KC alignment with `v1-CTA` (75 KCs) assessed on E-learning 2023

		Adj. Rand [-0.5, 1]	Adj. MI $(-\infty, 1]$	FM Index [0, 1]	Homogeneity [0, 1]	Completeness [0, 1]	V-measure [0, 1]
`Concept`	(298 KCs)	0.2888	0.6318	0.3511	**0.9377**	0.7092	**0.8076**
`Concept-emb`	(81 KCs)	0.3212	0.6091	0.3357	0.7384	0.7200	0.7291
`Question-emb`	(78 KCs)	0.3468	0.6385	0.3608	0.7535	0.7410	0.7472
`KCluster`	(92 KCs)	**0.4361**	**0.7077**	**0.4529**	0.8320	**0.7776**	0.8039

5.1.1 ScienceQA

Table 5 shows the results obtained from the ScienceQA dataset, where the "skill" tag of each MCQ serves as ground-truth labels. With far fewer KCs (198 vs. 549), `KCluster` consistently outperforms `Concept`, the method based on extracting concepts from questions, in all six measures, showing closer alignment with the gold standard `Skill` model. `Question-emb`, based on question embeddings, also surpasses `Concept` in all metrics except homogeneity, running closely after `KCluster`. We excluded `Concept-emb`, the method based on concept embeddings, because it did not converge after 200 iterations of affinity propagation.

The results on ScienceQA highlight that `Concept`, the most straightforward KC discovery method based on concept extraction using LLM, does not align with expert opinions better than the two clustering-based approaches, `KCluster` and `Question-emb`. Furthermore, `Concept` produces 4.5 times more KC labels than what is in the `Skill` model (549 vs. 99), which reaffirms the known limitation of this approach that it tends to produce excessive labels with word nuances. `KCluster`, however, generates an intermediate number of KCs and achieves the best score in four of the six metrics.

5.1.2 E-learning 2022

Table 6 shows the results obtained from the E-learning 2022 dataset, where `LOs-new`, the best expert-designed KC model according to Section 5.2.1, serves as the gold standard. With an intermediate number of KCs, `KCluster` leads the other three methods on almost every metric, except that `Concept` has better homogeneity and V-measure scores. A high homogeneity score indicates that `Concept` has many KCs containing questions that belong to the same KC in the `LOs-new` model, but does not take into account whether questions belonging to the same KC in `LOs-new` are always assigned to the same KC in `Concept`—in fact, for questions belonging to the KC "compare and contrast DFA and CTA skill" in the `LOs-new` model, `Concept` created five KCs, two of which read "a difficulty factors assessment" and "Difficulty Factors Assessment". While `Concept` produced redundant labels as discussed previously, it also created the least *complete* KC assignment where questions from the same ground-truth KC are scattered in multiple predicted KCs. In contrast, `KCluster` achieves the best completeness while maintaining the second-best homogeneity, despite marginally behind on the default V-measure that weighs both aspects equally.

5.1.3 E-learning 2023

Table 7 shows the results obtained from the E-learning 2023 dataset with `v1-CTA` as the gold standard. Since E-learning 2023 contains a subset of the questions in E-learning 2022, the results are consistent: `KCluster` outperforms all three other methods except that `Concept` has the best score in homogeneity and V-measure. `Concept` still created redundant KC labels and the least complete KC assignment—for a KC in `v1-CTA` about describing the redundancy principle in instructional design, `Concept` generated four KCs, three of which read "redundancy", "redundancy principle", and "the redundancy principle". To avoid redundant exposition, we conclude this section by highlighting that **with its lead on majority of the metrics, `KCluster` attained the best alignment with expert-designed KC models in all three datasets**.

5.2 Does KCluster enable accurate prediction of student responses? (RQ-2)

While `KCluster`'s close alignment with expert-designed KC models suggests that `KCluster` is a promising approach, fit to student performance data provides a more reliable benchmark. An effective KC extraction method should produce an informative KC model (in the form of a binary Q-matrix [1]) that an instructional expert can use with a statistical model to accurately predict student responses to questions. Our RQ-2 explores whether `KCluster` enables accurate student modeling, and if so, whether it outperforms the other methods. Using the student activity data from the E-learning 2022 and 2023 datasets, we train an Additive Factors Model (AFM) [3] with the generated Q-matrices to evaluate the predictive power of each KC extraction method and report the standard metrics of model fit used by DataShop.

AFM [3] is a logistic regression model that explains a student i's correct (1) or incorrect (0) response to a question j using the student's proficiency θ_i along with the KC difficulty β_k,

Table 8: AFM performance on E-learning 22 (50 CV runs)

		AIC	BIC	Item-RMSE (Std.)
`Single-KC`	(1 KC)	46227.9805	46582.6144	0.4264 (0.0002)
`Unique-step`	(1,865 KCs)	43323.0595	75923.4268	0.4273 (0.0002)
`LOs`	(87 KCs)	43972.6766	45815.0429	0.4244 (0.0010)
`LOs-new`	(101 KCs)	43353.2793	**45437.8345**	0.4236 (0.0016)
`Concept`	(371 KCs)	**41994.9029**	48750.2457	0.4295 (0.0017)
`Concept-emb`	(101 KCs)	44537.1400	46621.6952	0.4292 (0.0011)
`Question-emb`	(91 KCs)	43880.7030	45792.2660	0.4232 (0.0010)
`KCluster`	(114 KCs)	43424.5571	45734.0021	**0.4227** (0.0013)

Table 9: AFM performance on E-learning 23 (50 CV runs)

		AIC	BIC	Item-RMSE (Std.)
`Single-KC`	(1 KC)	46210.3867	46566.8170	0.4141 (0.0001)
`Unique-step`	(1,398 KCs)	42183.6839	66829.5327	0.4156 (0.0001)
`v1-CTA`	(75 KCs)	43434.4955	45077.5521	0.4088 (0.0021)
`v2-combined`	(72 KCs)	43471.4342	45062.3302	0.4088 (0.0024)
`Concept`	(298 KCs)	**41655.2518**	47175.5742	0.4111 (0.0021)
`Concept-emb`	(81 KCs)	44366.9480	46114.3256	0.4151 (0.0014)
`Question-emb`	(78 KCs)	43946.2607	45641.4778	0.4108 (0.0011)
`KCluster`	(92 KCs)	42999.9064	**44938.5393**	**0.4071** (0.0009)

Table 10: Improvements to the KC "11.1 apply_evidence" in E-learning 22 (50 CV runs)

		AIC	BIC	Item-RMSE (Std.)
`LOs-new`		43353.2716	45437.8268	0.4230 (0.0012)
`Concept`	(+5 KCs)	43265.7209	45419.4730	0.4223 (0.0012)
`Question-emb`	(+3 KCs)	43284.1397	45403.2933	0.4225 (0.0012)
`KCluster`	(+4 KCs)	**43262.7614**	**45399.2143**	**0.4221** (0.0013)

the KC learning rate γ_k, and the number of student practices T_{ik} for the relevant KCs as defined by a binary Q-matrix whose entry q_{jk} indicates if question j is associated with KC k. If Y_{ij} denotes a student i's response to a question j, AFM computes the log-odds of the student giving correct response ($Y_{ij} = 1$) as a linear combination of these factors:

$$\log \frac{\Pr(Y_{ij} = 1)}{\Pr(Y_{ij} = 0)} = \theta_i + \sum_k q_{jk}\beta_k + q_{jk}\gamma_k T_{ik} \tag{3}$$

Different KC extraction approaches tend to produce a different Q-matrix and thus instantiate a distinct AFM (via q_{jk}), for which the maximum likelihood estimation converges to different parameter estimates for θ_i, β_k, and γ_k, allowing us to compare different approaches. Following the standard practice in DataShop[8], we report the Akaike information criterion (AIC) and Bayesian information criterion (BIC), which describe how well an AFM fits the *current* data; moreover, we perform a cross-validation (CV) procedure that randomly divides questions (or *items*) into **three folds** and repeat it with **50 different random seeds** to report the average item-stratified root mean square error (item-RMSE), which predicts how well an AFM generalizes to *unseen* data. Stratifying the data by questions allows us to predict a student's responses to novel questions in the validation fold based on their responses to questions in the training folds, which is more relevant to our RQ-2. For all metrics, a lower value indicates a better prediction of student responses.

5.2.1 *E-learning 2022*

Table 8 summarizes the results on the E-learning 2022 dataset. In addition to fitting an AFM with the Q-matrix generated by each automated KC extraction method, we also fit an AFM with the Q-matrix obtained from the two default KC models (`Single-KC` and `Unique-step`) and the two expert-designed models (`LOs` and `LOs-new`) for comparison.

We observe that, although we did not use elaborate prompting strategies in our prompt template (Table 1), `Concept` is still a strong baseline with the best AIC among all models. The embedding-based approach, `Concept-emb`, managed to reduce the 371 KCs produced by `Concept` to 101 KCs via concept embedding and clustering, and consequently improved BIC, which favors models with fewer parameters. The other embedding-based approach `Question-emb`, however, outperforms `Concept-emb` in all metrics and achieves an item-RMSE comparable ($t(98) = -1.4738$, $p = .1437$) to that achieved by `LOs-new`, which has the best item-RMSE among expert KC models. This reinforces our initial prediction that encoding questions as embeddings should yield a better model than encoding concepts does, since questions contain more information than concepts do.

Using the novel question congruity to measure question similarity, `KCluster` outperforms all other automated KC extraction methods except for having a higher AIC than `Concept`. In particular, `KCluster` significantly exceeds the best expert-designed KC model, `LOs-new`, in item-RMSE ($t(98) = -2.9963$, $p = .0035$) at $\alpha = .05$. Compared to `Question-emb`, which measures similar questions using the traditional negative cosine distance, `KCluster` fits to the student data better as evidenced by better AIC and BIC scores, and is likely to predict unseen data more accurately as evidenced by a better item-RMSE ($t(98) = -2.1145$, $p = .0370$). Together, these results suggest that it is advantageous to identify clusters of similar questions and assign KCs to clusters (as done by `KCluster`) rather than to individual questions (as done by `Concept`), and that question congruity is more effective than negative cosine distance for measuring similar questions in clustering-based KC discovery.

5.2.2 *E-learning 2023*

Table 9 shows the results obtained from the E-learning 2023 dataset, where we also trained an AFM for the two expert models, `v1-CTA` and `v2-combined`. Although all questions in E-learning 2023 are also present in E-learning 2022, the activity data come from a different student cohort, allowing us to assess whether each method is robust against different students. Consistent with what is observed in E-learning 2022, `KCluster` leads all three other automated methods in almost every metric, only slightly behind `Concept` on AIC; it has the best BIC score among all models, manual or automated, indicating that `KCluster` fits the current data the best. The two expert models have comparable scores on all measures, but `KCluster` outperforms both models in AIC and BIC, and significantly so in item-RMSE ($t(98) = -5.0956$, $p < .001$). In addition, `KCluster` significantly outperforms `Question-emb` in item-RMSE ($t(98) = -18.1487$, $p < .001$), reaffirming our conclusion from E-learning 2022 that **question congruity is superior to negative cosine distance in measuring question similarity**.

[8] `https://pslcdatashop.web.cmu.edu/help?page=modelValues#values`

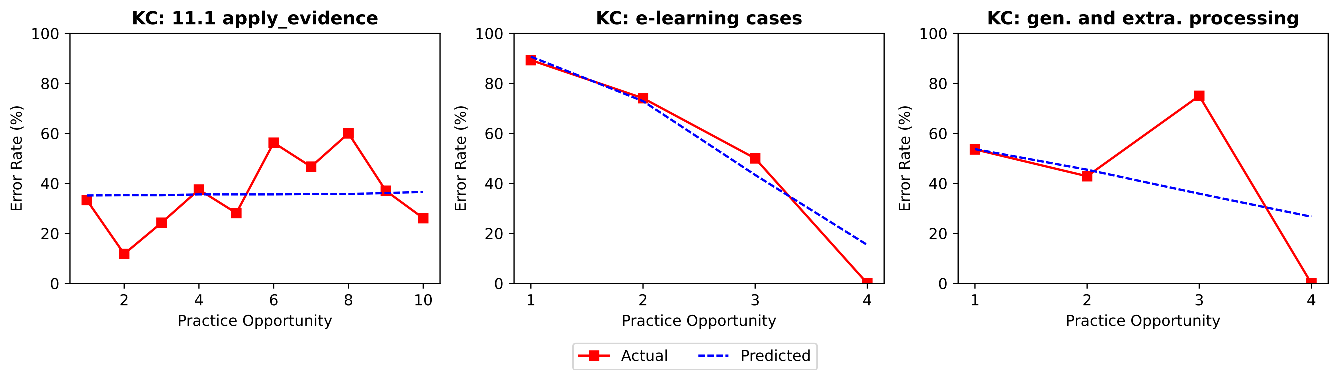


Figure 2: The learning curves for the original expert KC (left) and two new KCs discovered by `KCluster` (middle and right)

5.3 Does KCluster reveal insights about problematic KCs? (RQ-3)

By generating an alternate KC model, `KCluster` suggested how questions could have been organized by KCs so that an instructor can better predict student responses, but it did not explain, for example, why learning was difficult for some problematic KCs in the original expert KC model. A KC is problematic (and worth investigating) if it is neither too difficult nor too easy to learn, yet the students did not show any learning [46]. Previous work using data-driven DFA [46] manually analyzed and divided a problematic KC into three hidden KCs, which improved the prediction of student responses when reinserted into the original model. Our RQ-3 explores how `KCluster` can *automatically* reveal similar insights about and suggest improvements to problematic KCs.

From the E-learning 2022 dataset, we first identified 14 problematic KCs in the `LOs-new` model that an AFM estimated to have a learning rate $\gamma_k < 0.001$ (students were not learning) and an initial success probability (equal to $\texttt{sigmoid}(\beta_k)$) between 0.2 and 0.8 (the KC was neither too difficult nor too easy to learn). Following previous work [46], we then applied `Concept`, `Question-emb`, and `KCluster` to the questions associated with each problematic KC and discovered new KCs that constitute the original. We searched for improvement, where an AFM achieves a lower item-RMSE, in each new KC model that had a problematic KC replaced, and found that `Concept` and `KCluster` significantly improved the KC "11.1 apply_evidence", which has a zero learning rate and an initial success probability of 0.65.

Table 10 quantifies the improvements. Compared to `LOs-new`, all methods divide the original KC into multiple new KCs, suggesting that the expert KC is too coarse to reflect a single skill. `KCluster` breaks the generic "11.1 apply_evidence" KC into four different KCs, three of which concern specific sources of evidence ("generative and extraneous processing", "the practice or testing effect", and "e-learning cases"), with a fourth "evidence" KC for problems that contrast evidence and ask students to decide which situation would yield better learning. When reinserted into the original `LOs-new` model, the four new KCs discovered by `KCluster` brought the greatest improvements in all three metrics and significantly in item-RMSE ($t(98) = -3.4379, p < .001$).

This automated DFA not only discovers more specific and potentially more meaningful KCs, but also captures student learning better. Figure 2 contrasts the learning curve of the original "11.1 apply_evidence" KC with that of two new KCs ("e-learning cases" and "generative and extraneous processing") created by `KCluster`. While the original learning curve remains flat after ten learning opportunities, the error rates depicted in the new learning curves quickly approach zero after four opportunities, showing clear evidence of learning. An instructor, after reviewing the new learning curves, will be able to make informed adjustments to instruction and improve student learning *specifically* in the other two aspects of "applying evidence", with which students were struggling (namely, "the practice or testing effect" and "evidence"). This shows that **`KCluster` is not only capable of predicting student responses in foresight, but it can also illuminate improvements to instruction in retrospect**.

6. GENERAL DISCUSSION

Our comprehensive evaluation reveals three critical insights about `KCluster` that we will discuss in this section.

Clustering-based approaches outperform concept extraction. Using the text generation ability of Phi-2, `Concept` is a natural LLM-based method to extract KCs from questions. Yet, *using the same LLM*, `KCluster` shows that closer alignment with expert models (Section 5.1), better prediction of student responses (Section 5.2), and greater improvement to problematic KCs (Section 5.3) can be achieved by coupling Phi-2's native ability to evaluate text probabilities with clustering. That we chose Phi-2 over more advanced LLMs for Phi-2's balanced performance and affordability does not account for this performance discrepancy, as both methods use the same LLM. In fact, using a more advanced LLM and a curated set of 80 MCQs from the E-learning 2022 dataset, previous work [37] only managed to produce the exact KC for 28 MCQs (35%). A possible reason for this low KC match rate is that the powerful LLM generated redundant labels with undesired word nuances. Clustering-based approaches like `KCluster`, on the other hand, reduce the redundancy by propagating the labels of the cluster exemplars. As a rising tide will lift all boats, we expect future work using a more advanced LLM to improve both classes of methods, but Phi-2 is free and therefore more readily available to instructors.

Question congruity is more effective than negative cosine distance in measuring similar questions for clustering-based KC discovery. Both `KCluster` and `Question-emb` use affinity propagation [15] to identify clusters of similar questions and label all questions in a cluster with a KC equivalent to the concept label of the cluster exemplar. `KCluster`, however, outperforms `Question-emb` in aligning with expert-designed KC models, predicting student responses, and improving problematic KCs, by using the novel question congruity described in Section 3.3 (rather than the traditional negative cosine distance) to measure question similarity. These positive results have strengthened our belief that future work will prove question congruity a strong measure of question similarity in more domains than KC discovery.

Automated approaches can outperform manual approaches. Combining the strengths of LLM and clustering, `KCluster` enables instructors to predict student responses better than the best expert model does in the two e-learning datasets (Section 5.2). While we expect future work to extend `KCluster` to more datasets and more question types, our evaluation offers strong evidence that `KCluster`, an automated approach, can surpass manual approaches in modeling student learning. Furthermore, `KCluster` has demonstrated initial success in automated DFA (Section 5.3), inspiring future work that closes the loop by implementing and validating new instructional designs informed by `KCluster`.

7. CONCLUSION

We proposed question congruity, a novel measure of question similarity based on question collocations, and described an algorithm that uses Phi-2 to compute the required probabilities. The two contributions underlie `KCluster`, a novel KC discovery approach that combines LLM and clustering. Our comprehensive evaluation shows that `KCluster` not only outperforms the other three competing methods and the best expert KC model, but can also offer insights into problematic KCs that potentially inspire new instructional designs.

8. ACKNOWLEDGMENTS

This research was supported by the National Science Foundation under Grant No. 2301130 and a Google Academic Research Award to Paulo F. Carvalho.

This research was also supported by a US Navy grant on Real-time Knowledge Sharing awarded to John Stamper under Grant No. N68335-23-C-0035.

9. REFERENCES

[1] T. Barnes. The q-matrix method: Mining student response data for knowledge. In *AAAI Workshop*, 2005.

[2] H. Cen, K. Koedinger, and B. Junker. Learning factors analysis – a general method for cognitive model evaluation and improvement. In M. Ikeda, K. D. Ashley, and T.-W. Chan, editors, *Intelligent Tutoring Systems*, pages 164–175, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg.

[3] H. Cen, K. R. Koedinger, and B. Junker. Is over practice necessary? improving learning efficiency with the cognitive tutor through educational data mining. In *Proceedings of the 2007 Conference on Artificial Intelligence in Education: Building Technology Rich Learning Contexts That Work*, page 511–518, NLD, 2007. IOS Press.

[4] H. Chau, I. Labutov, K. Thaker, D. He, and P. Brusilovsky. Automatic Concept Extraction for Domain and Student Modeling in Adaptive Textbooks. *International Journal of Artificial Intelligence in Education*, 31(4):820–846, Dec. 2021.

[5] M. Chen, J. Tworek, H. Jun, Q. Yuan, H. P. d. O. Pinto, J. Kaplan, H. Edwards, Y. Burda, N. Joseph, G. Brockman, A. Ray, R. Puri, G. Krueger, M. Petrov, H. Khlaaf, G. Sastry, P. Mishkin, B. Chan, S. Gray, N. Ryder, M. Pavlov, A. Power, L. Kaiser, M. Bavarian, C. Winter, P. Tillet, F. P. Such, D. Cummings, M. Plappert, F. Chantzis, E. Barnes, A. Herbert-Voss, W. H. Guss, A. Nichol, A. Paino, N. Tezak, J. Tang, I. Babuschkin, S. Balaji, S. Jain, W. Saunders, C. Hesse, A. N. Carr, J. Leike, J. Achiam, V. Misra, E. Morikawa, A. Radford, M. Knight, M. Brundage, M. Murati, K. Mayer, P. Welinder, B. McGrew, D. Amodei, S. McCandlish, I. Sutskever, and W. Zaremba. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, July 2021.

[6] K. W. Church and P. Hanks. Word association norms, mutual information, and lexicography. In *27th Annual Meeting of the Association for Computational Linguistics*, pages 76–83, Vancouver, British Columbia, Canada, June 1989. Association for Computational Linguistics.

[7] R. E. Clark, D. Feldon, J. J. G. van Merriënboer, K. Yates, and S. Early. Cognitive task analysis. In J. M. Spector, M. D. Merrill, J. J. G. van Merriënboer, and M. P. Driscoll, editors, *Handbook of research on educational communications and technology*, pages 577–593. Macmillan/Gale, New York, 3rd edition, 2008.

[8] K. Cobbe, V. Kosaraju, M. Bavarian, M. Chen, H. Jun, L. Kaiser, M. Plappert, J. Tworek, J. Hilton, R. Nakano, C. Hesse, and J. Schulman. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, October 2021.

[9] A. T. Corbett and J. R. Anderson. Knowledge tracing: Modeling the acquisition of procedural knowledge. *User Modeling and User-Adapted Interaction*, 4(4):253–278, Dec. 1994.

[10] J. de la Torre. Dina model and parameter estimation: A didactic. *Journal of Educational and Behavioral Statistics*, 34(1):115–130, 2009.

[11] M. C. Desmarais, B. Beheshti, and R. Naceur. Item to skills mapping: Deriving a conjunctive q-matrix from data. In S. A. Cerri, W. J. Clancey, G. Papadourakis, and K. Panourgia, editors, *Intelligent Tutoring Systems*, pages 454–463, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg.

[12] M. C. Desmarais and R. Naceur. A matrix factorization method for mapping items to skills and for enhancing expert-based q-matrices. In H. C. Lane, K. Yacef, J. Mostow, and P. Pavlik, editors, *Artificial Intelligence in Education*, pages 441–450, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg.

[13] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova.

Bert: Pre-training of deep bidirectional transformers for language understanding. *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, June 2019.

[14] E. B. Fowlkes and C. L. Mallows. A method for comparing two hierarchical clusterings. *Journal of the American Statistical Association*, 78(383):553–569, 1983.

[15] B. J. Frey and D. Dueck. Clustering by passing messages between data points. *Science*, 315(5814):972–976, 2007.

[16] J. P. González-Brenes and J. Mostow. What and when do students learn? fully data-driven joint estimation of cognitive and student models. In *Educational Data Mining*, 2013.

[17] S. Gunasekar, Y. Zhang, J. Aneja, C. C. T. Mendes, A. D. Giorno, S. Gopi, M. Javaheripi, P. Kauffmann, G. de Rosa, O. Saarikivi, A. Salim, S. Shah, H. S. Behl, X. Wang, S. Bubeck, R. Eldan, A. T. Kalai, Y. T. Lee, and Y. Li. Textbooks are all you need, 2023.

[18] R. Hambleton and H. Swaminathan. *Item Response Theory: Principles and Applications*. Springer Science+Business Media, New York, NY, USA, 1985.

[19] N. T. Heffernan and K. R. Koedinger. A developmental model for algebra symbolization: The results of a difficulty factors assessment. In *Proceedings of the Twentieth Annual Conference of the Cognitive Science Society*, pages 484–489, Mahwah, NJ, 1998. Lawrence Erlbaum Associates, Inc.

[20] M. Javaheripi, S. Bubeck, M. Abdin, J. Aneja, S. Bubeck, C. C. T. Mendes, W. Chen, A. D. Giorno, R. Eldan, S. Gopi, S. Gunasekar, M. Javaheripi, P. Kauffmann, Y. T. Lee, Y. Li, A. Nguyen, G. de Rosa, O. Saarikivi, A. Salim, S. Shah, M. Santacroce, H. S. Behl, A. T. Kalai, X. Wang, R. Ward, P. Witte, C. Zhang, and Y. Zhang. Phi-2: The surprising power of small language models, Dec. 2023.

[21] A. Q. Jiang, A. Sablayrolles, A. Mensch, C. Bamford, D. S. Chaplot, D. d. l. Casas, F. Bressand, G. Lengyel, G. Lample, L. Saulnier, T. Lavril, M.-A. Lachaux, D. Massiceti, J. Rio, R. Lambert, S. Bhosale, S. Aminov, W. Kool, R. Everett, A. Gu, S. Dukma, H. Hao, X. Zhou, J. Chen, C. Iovine, W. Chen, V. Wang, and J. Calandriello. Mistral 7b. *arXiv preprint arXiv:2310.06825*, October 2023.

[22] D. Jurafsky and J. H. Martin. *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition with Language Models*. 3rd edition, 2024. Online manuscript released August 20, 2024.

[23] K. R. Koedinger, A. T. Corbett, and C. Perfetti. The knowledge-learning-instruction framework: Bridging the science-practice chasm to enhance robust student learning. *Cognitive Science*, 36(5):757–798, 2012.

[24] K. R. Koedinger and E. A. McLaughlin. Seeing language learning inside the math: Cognitive analysis yields transfer. In S. Ohlsson and R. Catrambone, editors, *Proceedings of the 32nd Annual Conference of the Cognitive Science Society*, pages 471–476. Cognitive Science Society, 2010.

[25] K. R. Koedinger, E. A. McLaughlin, and J. C. Stamper. Automated Student Model Improvement. Technical report, International Educational Data Mining Society, June 2012. ERIC Number: ED537201.

[26] K. R. Koedinger and M. J. Nathan. The real story behind story problems: Effects of representations on quantitative reasoning. *Journal of the Learning Sciences*, 13(2):129–164, 2004.

[27] K. R. Koedinger, J. C. Stamper, E. A. McLaughlin, and T. Nixon. Using data-driven discovery of better student models to improve student learning. In H. C. Lane, K. Yacef, J. Mostow, and P. Pavlik, editors, *Artificial Intelligence in Education*, pages 421–430, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg.

[28] K. R. Koedinger, M. V. Yudelson, and P. I. Pavlik Jr. Testing Theories of Transfer Using Error Rate Learning Curves. *Topics in Cognitive Science*, 8(3):589–609, 2016.

[29] A. S. Lan, A. E. Waters, C. Studer, and R. G. Baraniuk. Sparse factor analysis for learning and content analytics. *Journal of Machine Learning Research*, 15(57):1959–2008, 2014.

[30] N. Li, W. W. Cohen, K. R. Koedinger, and N. Matsuda. A machine learning approach for automatic student model discovery. In M. Pechenizkiy, T. Calders, C. Conati, S. Ventura, C. Romero, and J. C. Stamper, editors, *Proceedings of the 4th International Conference on Educational Data Mining, Eindhoven, The Netherlands, July 6-8, 2011*, pages 31–40. www.educationaldatamining.org, 2011.

[31] N. Li, E. Stampfer, W. Cohen, and K. Koedinger. General and Efficient Cognitive Model Discovery Using a Simulated Student. *Proceedings of the Annual Meeting of the Cognitive Science Society*, 35(35), 2013.

[32] J. Liu, G. Xu, and Z. Ying. Data-Driven Learning of Q-Matrix. *Applied psychological measurement*, 36(7):548–564, Oct. 2012.

[33] R. Liu and K. R. Koedinger. Closing the loop: Automated data-driven cognitive model discoveries lead to improved instruction and learning gains. *Journal of Educational Data Mining*, 9(1):25–41, Sep. 2017.

[34] P. Lu, S. Mishra, T. Xia, L. Qiu, K.-W. Chang, S.-C. Zhu, O. Tafjord, P. Clark, and A. Kalyan. Learn to explain: Multimodal reasoning via thought chains for science question answering. In *The 36th Conference on Neural Information Processing Systems (NeurIPS)*, 2022.

[35] N. Matsuda, J. Wood, R. Shrivastava, M. Shimmei, and N. Bier. Latent skill mining and labeling from courseware content. *Journal of Educational Data Mining*, 14(2), Oct. 2022.

[36] R. Mihalcea and P. Tarau. TextRank: Bringing order into text. In D. Lin and D. Wu, editors, *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, pages 404–411, Barcelona, Spain, July 2004. Association for Computational Linguistics.

[37] S. Moore, R. Schmucker, T. Mitchell, and J. Stamper. Automated generation and tagging of knowledge components from multiple-choice questions. In *Proceedings of the Eleventh ACM Conference on*

Learning @ Scale, L@S '24, page 122–133, New York, NY, USA, 2024. Association for Computing Machinery.

[38] B. PaaÃŸen, M. Dywel, M. Fleckenstein, and N. Pinkwart. Sparse factor autoencoders for item response theory. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 17–26, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

[39] Z. A. Pardos and A. Dadu. dAFM: Fusing Psychometric and Connectionist Modeling for Q-matrix Refinement. *Journal of Educational Data Mining*, 10(2):1–27, Oct. 2018. Number: 2.

[40] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Köpf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala. *PyTorch: an imperative style, high-performance deep learning library*. Curran Associates Inc., Red Hook, NY, USA, 2019.

[41] N. Reimers and I. Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 11 2019.

[42] A. Rosenberg and J. Hirschberg. V-measure: A conditional entropy-based external cluster evaluation measure. In J. Eisner, editor, *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 410–420, Prague, Czech Republic, June 2007. Association for Computational Linguistics.

[43] S. J. Russell and P. Norvig. *Artificial Intelligence: A Modern Approach*. Pearson, 4th edition, 2020.

[44] K. M. Shabana and C. Lakshminarayanan. Unsupervised Concept Tagging of Mathematical Questions from Student Explanations. In N. Wang, G. Rebolledo-Mendez, N. Matsuda, O. C. Santos, and V. Dimitrova, editors, *Artificial Intelligence in Education*, pages 627–638, Cham, 2023. Springer Nature Switzerland.

[45] Y. Shi, R. Schmucker, M. Chi, T. Barnes, and T. Price. KC-Finder: Automated Knowledge Component Discovery for Programming Problems. Technical report, International Educational Data Mining Society, 2023. ERIC Number: ED630850.

[46] J. C. Stamper and K. R. Koedinger. Human-machine student model discovery and improvement using datashop. In G. Biswas, S. Bull, J. Kay, and A. Mitrovic, editors, *Artificial Intelligence in Education*, pages 353–360, Berlin, Heidelberg, 2011. Springer Berlin Heidelberg.

[47] J. C. Stamper, K. R. Koedinger, R. S. J. d. Baker, A. Skogsholm, B. Leber, S. Demi, S. Yu, and D. Spencer. Datashop: A data repository and analysis service for the learning science community (interactive event). In G. Biswas, S. Bull, J. Kay, and A. Mitrovic, editors, *Artificial Intelligence in Education*, pages 628–628, Berlin, Heidelberg, 2011. Springer Berlin Heidelberg.

[48] D. Steinley. Properties of the Hubert-Arable Adjusted Rand Index. *Psychological Methods*, 9(3):386–396, 2004.

[49] C. Tofel-Grehl and D. F. Feldon. Cognitive task analysis–based training: A meta-analysis of studies. *Journal of Cognitive Engineering and Decision Making*, 7(3):293–304, 2013.

[50] H. Touvron, L. Martin, K. Stone, P. Albert, A. Almahairi, Y. Babaei, N. Bashlykov, S. Batra, P. Bhargava, S. Bhosale, D. Bikel, L. Blecher, C. C. Ferrer, M. Chen, G. Cucurull, D. Esiobu, J. Fernandes, J. Fu, W. Fu, B. Fuller, C. Gao, V. Goswami, N. Goyal, A. Hartshorn, S. Hosseini, R. Hou, H. Inan, M. Kardas, V. Kerkez, M. Khabsa, I. Kloumann, A. Korenev, P. S. Koura, M.-A. Lachaux, T. Lavril, J. Lee, D. Liskovich, Y. Lu, Y. Mao, X. Martinet, T. Mihaylov, P. Mishra, I. Molybog, Y. Nie, A. Poulton, J. Reizenstein, R. Rungta, K. Saladi, A. Schelten, R. Silva, E. M. Smith, R. Subramanian, X. Tan, B. Tang, R. Thakoor, P. Trinh, T.-H. Tsai, X. Wang, W. Wang, Z. Wu, Y. Zhang, M. Zhang, P. Zheng, M. Zhou, and W. Zhu. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, July 2023.

[51] N. X. Vinh, J. Epps, and J. Bailey. Information theoretic measures for clusterings comparison: is a correction for chance necessary? In *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML '09, page 1073–1080, New York, NY, USA, 2009. Association for Computing Machinery.

[52] T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz, J. Davison, S. Shleifer, P. von Platen, C. Ma, Y. Jernite, J. Plu, C. Xu, T. Le Scao, S. Gugger, M. Drame, Q. Lhoest, and A. Rush. Transformers: State-of-the-art natural language processing. In Q. Liu and D. Schlangen, editors, *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online, Oct. 2020. Association for Computational Linguistics.

[53] Y. Wu, M. Schuster, Z. Chen, Q. V. Le, M. Norouzi, W. Macherey, M. Krikun, Y. Cao, Q. Gao, K. Macherey, J. Klingner, A. Shah, M. Johnson, X. Liu, Łukasz Kaiser, S. Gouws, Y. Kato, T. Kudo, H. Kazawa, K. Stevens, G. Kurian, N. Patil, W. Wang, C. Young, J. Smith, J. Riesa, A. Rudnick, O. Vinyals, G. Corrado, M. Hughes, and J. Dean. Google's neural machine translation system: Bridging the gap between human and machine translation, 2016.