Targeted Feedback Generation for Constructed-Response Questions

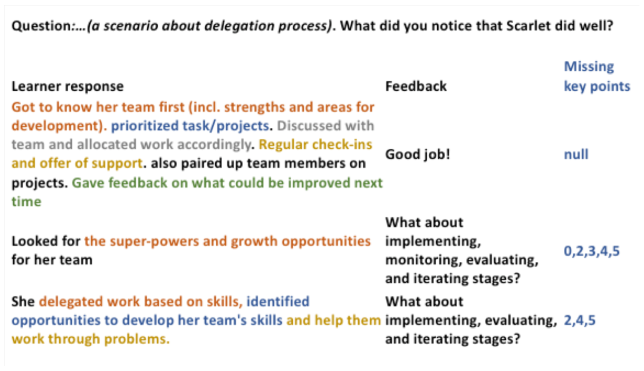
Jinjin Zhao,[1] Kim Larson, [1] Weijie Xu, [1] Neelesh Gattani, [1] Candace Thille [1]
[1] Amazon.com
jinjzhao, kilarson, weijiexu, neeleshg, cthille@amazon.com

Abstract

Constructed-response questions (CRQs) are an important activity that can help foster generative processing and promote a deeper understanding of the core content for learners. However, providing feedback and grading free-form text responses is labor intensive. This paper proposes a novel solution for providing targeted feedback automatically in online learning environments without any model training process. We leverage human-defined model answers and grading rubrics to generate feedback for the learner's answer. We apply state-of-the-art natural language processing (NLP) techniques, including text segmentation, pretrained language models (LMs), and contextualized text embeddings, to discover the misconceptions and provide feedback based on the misconceptions. We demonstrate the proposed solution with two CRQs embedded in an open-navigation online learning system that is focused on workforce learning. We measure the accuracy of the machine-generated feedback with human expert annotations. We use true positive rate (TPR) and false positive rate (FPR) as the statistical measure to validate the accuracy of the models. We report that semantic-based key phrase extraction outperforms statistics-based and graph-based approaches. We also report that pretrained LMs fine-tuned on similar tasks achieve the best performance compared to noncontextualized embedding approaches. To the best of our knowledge, this is the first work in evaluating the accuracy of misconception (interchangeable with knowledge gap, missing/inaccurate key points) analysis. We establish a baseline with 75% TPR and 22% FPR with semantic-based key phrase extraction and contextualized embedding approaches. We also demonstrate that semantic-based segmentation could potentially reduce human efforts in designing the comprehensive grading rubrics.

Introduction and Related Work

Constructed-response questions (CRQs) are an important approach to assess a learner's skills, such as the ability to organize thoughts, provide explanations, and demonstrate higher-level thinking skills. It requires more elaborate answers and explanations of reasoning (Clark and Mayer 2016). One type of CRQ is a short-answer format that requires learners to write one or several sentences in response



Question:*...(a scenario about delegation process)***. What did you notice that Scarlet did well?**

Learner response	**Feedback**	**Missing key points**
Got to know her team first (incl. strengths and areas for development). prioritized task/projects. Discussed with team and allocated work accordingly. Regular check-ins and offer of support. also paired up team members on projects. Gave feedback on what could be improved next time	Good job!	null
Looked for the super-powers and growth opportunities for her team	What about implementing, monitoring, evaluating, and iterating stages?	0,2,3,4,5
She delegated work based on skills, identified opportunities to develop her team's skills and help them work through problems.	What about implementing, evaluating, and iterating stages?	2,4,5

Figure 1: Example of one CRQ with learner answers. Missing/inaccurate key points are analyzed by comparing learner answers with the model answer. Feedback is provided on the basis of misconceptions. Model answer has six key points as follows: 1. Prioritize tasks and projects explicitly. 2. Consider her employee's capabilities. 3. Clearly communicate when assigning tasks. 4. Follow up on status of what she has delegated. 5. Evaluate her delegation process and employee actions. 6. Iterate on her delegation process to improve for the next time. Model answer includes five aspects as follows: planning, implementing, monitoring, evaluating, and iterating.

to an open-ended prompt. Because providing feedback and grading is difficult and labor intensive, researchers have been working on automatic short-answer grading (ASAG) since early 2000. Regression-based approaches (Sultan, Salazar, and Sumner 2016) have shown their high accuracy by taking advantage of supervised learning. The shortcoming of such approaches is that it requires an adequate amount (hundreds or thousands, dependent on the specific applications and required performance) of labeled data for the machine learning algorithm to capture the grading patterns. Another set of solutions (Basu, Jacobs, and Vanderwende 2013; Süzen et al. 2020; Lan et al. 2015) is to cluster learner responses into groups so that humans can intervene in the process to design rubrics for each group. With the defined rubrics, the upcoming learner answers can be graded automatically by the machine learning algorithm. The shortcoming is that it requires human efforts to interpret the clustering results from the machine learning algorithm and design rubrics accordingly. The interpretation process is even more difficult if the intended instructional purpose is complex and the learner behavior is diverse. There are continuous efforts on measuring syntactic (Leacock and Chodorow 2003) and semantic similarity between a learner's answer and a model answer, and grading based on the similarity. Mohler and Mihalcea reported knowledge-based and corpus-based approaches in representing the semantics (Mohler and Mihalcea 2009). Selvi and Bnerjee simply applied word matching in measuring the answer similarity (Selvi and Bnerjee 2010). Saha and colleagues proposed to match the learner answer to the model answer with sentence embedding (Saha et al. 2018) . Mohler, Bunescu, and Mihalcea also proposed to grade short answers using semantic similarity measures and dependency graph alignments (Mohler, Bunescu, and Mihalcea 2011). ASAG can be helpful in grading learners' answers with binary or scaled scores, but it cannot give feedback on the knowledge gap/misconception a learner has when his/her score is low or zero.

In some cases of CRQ, we give learners an expert answer in place of any scoring or evaluation of their answers. For these examples, it is expected that learners will evaluate their answer compared to the expert answer. In addition, Frost argued that the model answer has its shortcoming as formative feedback, and explanation feedback is more effective than correct answer feedback (Frost 2016; Butler, Godbole, and Marsh 2013). Also, Huxman argued that personalized feedback is more effective than model answers (Huxham 2007). It has been argued that targeted feedback can help learners better evaluate their own answers and maximize the value of a CRQ activity. Thus, in this work, other than providing the model answer as feedback directly, we propose to dive into the learner answer and provide targeted feedback based on the misconceptions. We also aim to provide feedback even for the first group of learners (who interact with the learning experience in a cold start setting) where we haven't yet collected any data from learner answers. After we have some learner data, we create and incorporate grading rubrics into a model answer and evaluate its usefulness with different techniques.

In this work, we propose a systematic solution for generating targeted feedback for a CRQ activity at key point level. A point is a word, a phrase, or a sentence learners use to describe what they observe from or think about the given context in response to the CRQ. A key point is a point that can accurately describe some part of the given context. A model answer consists of all the key points for that CRQ. A missing or inaccurate key point is what learners fail to capture from the given context. The proposed solution includes an NLP pipeline and human-defined model answer and grading rubics (optional). Figure 1 illustrates the general idea of providing targeted feedback given a learner answer and the model answer. In this example, there are six key points in the model answer that we expect to see in learner answers. Misconceptions are analyzed by comparing the learner answer with the model answer. Feedback is generated by mapping the misconceptions with key points in the model answer. The main contributions of this work are as follows:

- An effective NLP pipeline for extracting key points from model answer and learner answers
- Leveraging key points in the model answer or grading rubrics for misconception analysis
- Leveraging learner answers for designing grading rubrics and incorporating grading rubrics into model answer
- Establishing an accuracy baseline of misconception analysis with TPR 75% and FPR 22%
- Insights on effectiveness of different NLP techniques in key phrase extraction and semantic embedding for misconception analysis
- Insights on effectiveness of grading rubrics with different NLP techniques

Approach

The targeted feedback generation approach takes in a learner answer, compares it with the model answer at key point level, derives the misconceptions, and generates feedback with a predefined template. The feedback can then be provided to learners to help them practice. We focus on misconception analysis in this work. First, we analyze the model answer and extract key points. After that, we map the key points to the learner answers and derive the missing or inaccurate key points. In addition, we leverage learner answers to design grading rubrics. We also incorporate grading rubrics into model answer and test the effectiveness of the grading rubric with different NLP techniques. In misconception analysis, we apply three classes of text segmentation approaches for analyzing both learner answer and model answer. We leverage two classes of semantic embedding approaches to capture the semantics of the text.

Prepare model answer and grading rubrics

A model answer is how an expert would answer the question. It is a correct answer that covers all the key points. A grading rubric is a criterion-based scoring guide consisting of a fixed measurement scale and descriptions of each score point's characteristics. Rubrics describe degrees of quality, proficiency, or understanding along a continuum (Wiggins, Wiggins, and McTighe 2005). First, we collect the data from learners to find samples or performance that illustrates our desired learner understanding or proficiency. Then, we create an analytic rubric using that data. We use Arter and McTighe's (Arter and McTighe 2001) six-step process for analyzing learner performance as a guideline. We incorporate the rubrics into the model answer for misconception analysis.

Answer segmentation

In order to provide targeted feedback, we first need to understand the learner answer and model answer and segment them into key points. In this step, we apply three classes of text segmentation approaches (statistics-based, graph-based, and semantic-based) to extract key points. For each class of

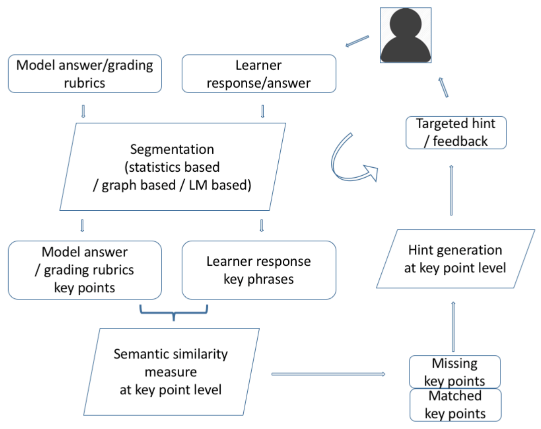

Model answer/grading rubrics
Learner response/answer
Segmentation (statistics based / graph based / LM based)
Targeted hint / feedback
Model answer / grading rubrics key points
Learner response key phrases
Hint generation at key point level
Semantic similarity measure at key point level
Missing key points
Matched key points

Figure 2: Targeted feedback generation framework. It takes in a learner answer and model answer as input and segment both answers with three classes of segmentation approaches. Rubrics is additionally designed and incorporated into model answer. Semantic embedding and similarity measure are conducted for each segment pair (learner input segment, model answer segment) with different embedding techniques. Missing/inaccurate key points are derived, and feedback is provided based on the misconceptions.

.

the approaches, we test two to three different algorithms to evaluate its effectiveness in the following experiments.

Statistics-based key phrase extraction Statistics-based methods extract key phrases based on the statistics (word frequency, word count, etc.) of the key phrase. Term frequency–inverse document frequency ($TFIDF$) is the most common statistics-based method. The method ranks score using the formula $TFIDF = TF \times IDF$, where TF stands for term frequency and IDF stands for inverse document frequency. In IDF, $1 + \log_2 \frac{N}{1+df_t}$, N is the number of documents and df_t is document frequency of the term. Although $TFIDF$ is effective in ranking important words, it does not keep track of position and context information. To capture context information, YAKE! (Campos R. 2018) is a feature-based system for multilingual keyword extraction from single documents. It uses casing W_{case} (casing aspect of a word), word position W_{pos} (words occurring at the beginning of a document), word frequency W_{freq}, word relevancy to context W_{rei} (computing the number of different terms that occur to the left/right side of the candidate word), and word difference in sentence W_{difs} (quantifying how often a candidate word appears within different sentences). The score is computed by $S(w) = \frac{W_{rei} * W_{pos}}{W_{case} + \frac{W_{freq}}{W_{rei}} + \frac{W_{difs}}{W_{rei}}}$, where *w* is the candidate word. For 2-gram and 3-gram, the score is assigned $S(kw) = \frac{\prod_{w \in kw} S(w)}{Tf(kw) * (1 + \sum_{w \in kw} S(w))}$, where *kw* is the candidate 2-gram or 3-gram. The smaller the score, the more meaningful the keyword will be. However, YAKE! in general needs large texts to perform well compared to $TFIDF$.

Graph-based key phrase extraction The basic idea in graph-based ranking is to create a graph from a document that has the candidate key phrases from the document as nodes and connections between candidate key phrases as edges. TextRank (Mihalcea and Tarau 2004) is the first graph-based key phrase extraction method. The text is first tokenized and annotated with POS (part-of-speech) tags. The method keeps only adjectives and nouns. Edges represent co-occurrence, and nodes represent those kept words. The initial score for each word is 1. Then, the score is updated by the formula $S(V_i) = 1 - \lambda + \lambda \cdot \sum_{j \in N(V_i)} \frac{S(V_j)}{|N(V_j)|}$, where $S(V_i)$ is the set of neighbors of V_i, $N(V_j)$ is the sum score of neighbors of V_j, and λ is the probability of jumping from one node to another. It runs until convergence. SingleRank (Mihalcea and Tarau 2008) is an extension of TextRank that uses co-occurrences of the two corresponding words to represent weights. The score function is updated in a similar way. In general, TextRank and SingleRank exploit intrinsic structural properties of text and find keywords that appear "central" and do not consider contextualized information. Rapid Automatic Keyword Extraction (RAKE) (Rose and Cowley 2010) is a domain-independent keyword extraction algorithm that tries to determine key phrases in a body of text by analyzing the frequency of word appearance and its co-occurrences with other words in the text. It utilizes both word frequency and word degree to assign scores to phrases. RAKE takes a list of stop words to partition a word into candidate phrases. Score is assigned on the basis of word-to-word co-occurrences. The score of phrases is calculated by the sum of the scores of the words in the phrase. RAKE can also identify key phrases that contain interior stop words. Finally, the top-ranked candidate phrases are selected as key phrases for the document. RAKE can generate more complicated phrases and its performance relies on quality of stop words. Multipartite Rank (Boudin 2018) builds a graph representation of the document, on which it applies a ranking algorithm to assign relevance score to each key phrase. Then it creates an in-between step where edge weights are adjusted to capture position information. This method gives higher weight to candidates that occur at the beginning of the document. It is claimed that it can achieve the highest MAP among topic-based methods. RaKUn (Blaž Škrlj 2019) exploits graph-based language representations for efficient denoising and keyword detection. It transforms texts into graph, maps words with similar lemma into same node, and extracts keywords on the basis of node importance. It works better than traditional graphic keyword extraction methods because the words-mapping part considers spelling errors, spelling differences, and non-standard writing.

Semantic-based key phrase extraction Semantic-based LMs are the state-of-the-art representation of text. We test key phrase extraction with pretrained LMs. We leverage two types of pretrained BERT models, basic (BERT-base-uncased) and BERT fine-tuned (DeepPavlov/BERT-base-cased-conversational), that have been proven effective in different NLP applications. BERT-base-uncased is a pretrained model on English language using a masked language modeling (MLM) objective. It does not make a difference between english and English (uncased). DeepPavlov/BERT-base-cased-conversational is a conversational BERT that is trained on the English part of Twitter, Reddit, Facebook News Comments, and several other public data sources. It uses this training data to build the vocabulary of English subtokens and takes the English cased version of BERT-base as an initialization for English Conversational BERT. We first extract potential phrases from the paragraph with segmentation techniques (tokenization, stemmation, lemmatization, etc.), and then we compare each phrase embedding with paragraph-level embedding. We hypothesize that the most semantically similar phrases (compared to paragraph semantics) are the key phrases of the given paragraph. After extracting the most relevant key phrases, we apply a diversity-based maximum marginal relevance (MMR) (Carbonell and Goldstein 1998) ranking algorithm to remove the duplicates.

Semantic embedding

After deriving the key points from the model answer and segmenting learner answers into points, we apply semantic matching to identify the matched key points and missing/inaccurate key points for the learner answer. We leverage both transformer-based pretrained LMs (contextualized) and non-transformer-based embedding for representing the text. For transformer-based LMs, we test sentence embedding with sentence transformer (Reimers and Gurevych 2019) and different BERT basic (BERT-base-cased, BERT-large-uncased) and Bert/Roberta fine-tuned LMs (distilbert-base-nli-mean-tokens, roberta-large-nli-stsb-mean-tokens). Basic LMs are purely trained with one certain objective (e.g., MLM for Bert). Fine-tuned LMs are basic LMs fine-tuned with specific tasks (e.g., semantic textual task for the two mentioned fine-tuned LMs). As researchers (Reimers 2020) summarized, distilbert-base-nli-mean-tokens is the most effective LM for a majority of the sentence-embedding applications. All the remaining LMs are the top-ranking recommended LMs. For non-transformer-based embedding, we leverage multitask convolutional neural network (CNN) (Ruder 2017) and GloVe (Jeffrey Pennington 2014) embedding. Multitask CNN leverages CNN architecture to train the word embedding on multiple tasks with co-occurrence statistics as input. GloVe is an unsupervised word-embedding methodology that uses co-occurrence statistics for embedding training. One of the biggest differences between transformer-based and traditional approaches is whether or not the text embedding is contextualized through a massive pretraining process.

Targeted feedback for nudging

Once we derive the missing/inaccurate key points from the semantic matching, we provide feedback to learners based on that result. We map the points from the learner answer to key points from the model answer with a multiple-to-one mapping and then generate feedback based on the misconceptions. We use an OR operation in evaluating each grading point for the set of key points that is mapped to. For example, we have five points [pt1, pt2, pt3, pt4, pt5] extracted from the learner answer and three key points [gp1, gp2, gp3] designed in the model answer. The mapping is defined as [pt1, pt2]-gp1, [pt3, pt4]-gp2, [pt5]-gp3. If a learner is able to touch upon two points, [pt1, pt3], then we would provide feedback for gp3 because that is the missing key point. After that, we construct a sentence using a predefined template. As Figure 1 shows, the template is defined as "What about..." stages. With the misconceptions, we form a final sentence,"What about Evaluating stage?" and communicate that to the learner after his/her attempt.

Evaluation

We evaluate model performance by measuring the accuracy of the misconceptions. We annotate a sample set of learner answers with human grading at key point level. Each key point is graded with a binary score. Two or three experts/course designers are asked to carefully grade the sample set of learner answers until a consensus is built among experts on each of the graded answer. We evaluate the model accuracy with true positive rate (TPR) and false positive rate (FPR) scores. TPR is the number of key points that are covered by learners. FPR is the number of key points that are not covered by the learners but are marked as covered by the model. We report the accuracy in an aggregated manner across learners.

Experiments and results

We conduct five experiments to evaluate different segmentation and semantic embedding approaches. We also test whether creating grading rubrics (from learner answer data) would improve model performance. We use two CRQs embedded in an open-navigation system to demonstrate the misconception analysis. We use expert annotation to evaluate the accuracy of the misconception analysis.

Experiment set up

CRQ activity introduction We use two open-ended questions from two different workforce training courses for managers. The first question is followed by a scenario of two managers with very different delegation processes, one delegating successfully and the other unsuccessfully. CRQ One, "What did you notice that Scarlet did well?" expects learners to articulate the best practices of delegation in the scenario. A model answer is provided to the algorithm. With learner data, a grading rubric is also created and incorporated into the model answer. The second question, CRQ Two, is followed by a scenario describing a manager's decision-making process: "What possible problems do you foresee occurring in Amber's situation?" A list of keyword candidates is prepared as the model answer. We simplify the grading with a binary decision. If a learner answer contains either of the keywords, that is a successful attempt. A rubric is also created using learner data. We sample 30 learners for each question for model evaluation. The sampled answer length for CRQ One ranges from 1 word to 8 sentences (150 words) with an average length of 50 words. The sampled answer length for CRQ Two ranges from 1 word to 80 words with an average length of 22 words.

Results and Insights

Experiment One. Learner answer segmentation We provided experimental results for CRQ One with different segmentation approaches for learner answers. For hyperparameters, top N phrases (based on n-gram) is an important hyperparameter because it would overemphasize a word by extracting it multiple times (as a single word or as part of an n-gram phrase). Similarity thresholding is also an important hyperparameter that is dependent on the choice of semantic embedding approach. We set hyperparameters as follows: 1. Top key phrase percentage for all the methods = 100%. 2. Semantic similarity thresholding = 85%. As shown in Table 1 (lines 1–4), across all four semantic embedding approaches, BERT basic and BERT fine-tuned perform better than pure statistics-based and graph-based approaches, where BERT-based approaches achieve a relative balance between TPR and FPR scores. Other approaches either have both high TPR and FPR (MultipartiteRank, RAKE) or low TPR and high FPR (POS, TFIDF, TextRank). It indicates that semantic-based segmentation benefits from understanding the semantics and could segment sentences into relatively meaningful chunks.

We also test different hyperparameters and see how much benefit we can achieve by refining the hyperparameters. We test top N phrases with a percentage of its total phrases, ranging from 50% to 100%, with 10% as an interval. We set similarity thresholding options to be 85%, 88%, 90%. We use the best performer, distilbert-base-nli-mean-tokens, as the LM (which will be demonstrated as the best option in Experiment Two). As Table 2 shows, a higher percentage of the top phrases will result in higher TPR and FPR, where more n-gram phrases are extracted. Lower similarity thresholding will result in higher TPR and FPR, where the definition of similar is relaxed. For example, RAKE with similarity thresholding as 85%, the best TPR (79%) is achieved when the top phrase percentage is set as 100%, where FPR ends up with 51%. If we restrict the relevancy and reduce the chance of overemphasizing with top phrase percentage set as 90%, we obtain 73% as TPR and 42% as FPR, which results a 6% decrease in TPR and a 9% improvement in FPR. If we restrict the definition of similar to 88%, we could obtain a relatively balanced result (64% TPR and 19% FPR). After fine-tuning, we obtain a balanced result for RAKE (15% TPR decrease as a cost and 32% FPR improvement). We also get a better result for the TFIDF approach, 73% TPR (3% decrease as a cost) and 37% FPR (11% improvement). From this set of experiments, we conclude that semantic-based key phrase extraction outperforms other approaches with a balanced result for learner answer segmentation. For statistics-based and graph-based key phrase extraction, we can fine-tune hyperparameters to achieve a balanced result.

Experiment Two. Semantic embedding We apply both transformer-based (contextualized embedding) and non-transformer-based approaches for semantic embedding. We extract key phrases of the model answer with TextRank segmentation. We extract key phrases of learner answers with POS, TFIDF, RAKE, TextRank, BERT basic, and BERT fine-tuned approaches. For transformer-based approaches, we apply BERT basic (BERT-base-cased, bert-large-uncased), BERT fine-tuned (distilbert-base-nli-mean-tokens), and Roberta fine-tuned (roberta-large-nli-stsb-mean-tokens) as LMs. Parameters are set as follows: 1. Top key phrase percentage for all the methods = 100%. 2. Semantic similarity thresholding = 85%. As Table 1 shows, LM (distilbert-base-nli-mean-tokens) optimized with a semantic textual similarity task (an NLP task that deals with determining how similar two pieces of texts are) outperforms the other three LMs in every segmentation setting considering the balance between TPR and FPR measure. For segmentation approaches, RaKUn and Multipartite Rank achieve the best TPR across all the LM models. TextRank, BERT basic, and BERT fine-tuned models achieve better FPR scores. For non-transformer-based approaches, we apply multitask CNN (trained on OntoNotes) and GloVe (1.2m 300d vectors trained on Common Crawl) as our embedding methods. We set hyperparameters as follows: 1. Top key phrase percentage for all the methods = 50%. 2. Semantic similarity thresholding = 40% for multitask CNN and 80% for GloVe. As Table 1 shows, multitask CNN outperforms GloVe in every segmentation setting. Across segmentation approaches, RAKE achieves the best FPR across all models. TFIDF and Multipartite Rank models achieve better TPR scores. Considering both TPR and FPR scores, graph-based methods (Rake and TextRank) perform better than the statistics-based (TFIDF) approach with multitask CNN as the embedding. The best performer of BERT fine-tuned outperforms multitask CNN and GloVe.

We achieve better results as we optimize parameters for non-transformer-based approaches. When we change the top key phrase percentage to 30% and semantic similarity threshold to 48% for multitask CNN, Multiparite Rank can achieve 78% TPR and 31% FPR. When we change the top key phrase percentage to 75% and semantic similarity threshold to 48% for GloVe, RAKE can achieve 75% TPR and 36% FPR. From this set of experiments, we observe that non-transformer-based embedding methods (that leverage a co-occurrence matrix) performs well with key phrase extraction (graph-based) that also leverages a co-occurrence matrix. In general, multitask CNN and GloVe work well when the hyperparameters are optimized. To summarize, BERT fine-tuned (on semantic textual similarity task) semantic embedding outperforms BERT basic, Roberta fine-tuned, and non-transformer-based approaches. In other words, fine-tuning an LM with similar tasks can improve the text contextualization. Non-transformer-based, multitask CNN, and GloVe could also work well when the hyperparameters are optimized for specific tasks.

Experiment Three. Model answer segmentation In this experiment, we conduct model answer segmentation with different approaches with CRQ One. For learner answer key phrase extraction, we apply BERT fine-tuned segmentation (the best performer concluded from Experiment One). For semantic embedding, we use distilbert-base-nli-mean-

	POS		TFIDF		RAKE		TextRank		BERT basic		BERT tuned		MultiP rank		RaKUn	
	TPR	FPR	TPR	FPR	TPR	FPR	TPR	FPR	TPR	FPR	TPR	FPR	TPR	FPR	TPR	FPR
1	0.64	0.55	0.65	0.53	0.69	0.55	0.57	0.33	0.56	0.38	0.56	0.37	0.9	0.63	**0.82**	**0.45**
2	0.65	0.48	0.67	0.50	0.67	0.50	0.43	0.34	0.50	0.30	0.49	0.30	0.46	0.17	**0.82**	**0.45**
3	0.71	0.52	0.76	0.48	**0.79**	**0.51**	0.64	0.37	**0.69**	**0.33**	**0.70**	**0.32**	**0.88**	**0.53**	**0.81**	**0.41**
4	0.16	0.01	0.05	0.02	0.15	0.02	0.09	0.02	0.15	0.01	0.15	0.01	0.19	0.02	0.42	0.11
5	0	0	0.86	0.53	0.75	0.36	0.77	0.44	0.26	0.16	0.96	0.75	**0.76**	**0.31**	0.64	0.36
6	0.88	0.62	0.16	0.24	**0.68**	**0.33**	0.68	0.40	0.58	0.27	0.18	0.08	0.27	0.28	0.23	0.16

Table 1: Learner answer segmentation and semantic embedding. 1: BERT-base-cased. 2: BERT-large-uncased. 3: distilbert-base-nli-mean-tokens. 4: roberta-large-nli-stsb-mean-tokens. 5: spacy (multi-task CNN). 6: spacy (GloVe). BERT basic and BERT fine-tuned key phrase extraction performs the best. For semantic embedding, BERT fine-tuned (3) outperforms others by contextualizing the text.

Accuracy	Top 50%		Top 60%		Top 70%		Top 80%		Top 90%		Top 100%	
	TPR	FPR	TPR	FPR	TPR	FPR	TPR	FPR	TPR	FPR	TPR	FPR
RAKE 85%	0.58	0.21	0.63	0.25	0.68	0.34	0.71	0.39	0.73	0.42	0.79	0.51
0.88 %	0.38	0.08	0.50	0.11	0.58	0.14	0.61	0.16	**0.64**	**0.19**	0.65	0.31
RAKE 90%	0.21	0.04	0.30	0.05	0.38	0.06	0.41	0.07	0.52	0.10	0.53	0.16
TFIDF 85%	0.60	0.32	0.63	0.34	0.67	0.37	0.69	0.37	**0.73**	**0.37**	0.76	0.48
TFIDF 88%	0.30	0.14	0.32	0.17	0.38	0.18	0.41	0.18	0.43	0.22	0.49	0.28
TFIDF 90%	0.16	0.06	0.16	0.08	0.21	0.08	0.23	0.09	0.27	0.10	0.31	0.13
TextRank 85%	0.54	0.18	0.56	0.22	0.56	0.24	0.58	0.26	0.58	0.26	0.64	0.37
TextRank 88%	0.32	0.02	0.34	0.05	0.34	0.06	0.36	0.07	0.38	0.07	0.41	0.08
TextRank 90%	0.17	0.01	0.20	0.02	0.21	0.03	0.23	0.03	0.23	0.03	0.24	0.04

Table 2: Model performance with different parameter settings for key phrase segmentation (non-semantic-based) for learner answers. Both statistics-based and graph-based methods can achieve decent performance after optimizing the parameter setting.

tokens (the best performer concluded in Experiment Two). We set the top key phrase percentage as 100% for all the methods. We have semantic similarity thresholding set as 85% or 88%. As Table 3 shows, most of the key phrase extraction approaches result in decent performance without massive fine-tuning. TFIDF, RAKE, and TextRank approaches achieve high performance with TPR around 70% and FPR around 22%. BERT-based approaches can even improve TPR to 75% with FPR around 25%. TFIDF and RAKE achieve better results when the semantic similarity thresholding is set as 88%. BERT basic and fine-tuned segmentation achieve better results with the thresholding set as 85%. To conclude, model answer segmentation is less sensitive to the key phrase extraction approach, and each of them could achieve a decent performance. Combining Experiment One, it indicates that if the text is well written (model answer is prepared following best writing practices, while learner answers are usually more casual), then non-semantic-based segmentation could also achieve decent performance.

Experiment Four. With grading rubrics The first three experiments are conducted given a model answer. In this experiment, we incorporate grading rubrics into model answer and evaluate its usefulness. We sampled a set of learner answers (30 entries) and manually designed the rubrics based on the answer patterns. All the experiment settings are the same as in Experiment Three. As shown in Table 3, TFIDF, RAKE, and TextRank achieve 5%, 10%, and 11% increase in TPR, respectively, after incorporating the grading rubrics. However, FPR is also increased by the same amount (performance decrease). BERT basic and BERT fine-tuned segmentation have stable performance for both TPR and FPR. It indicates that semantic-based key phrase extraction benefits from contextualizing the text representation. Regardless the length of the model answer or the amount of key phrases provided as the model answer (as long as the model answer has provided the semantics of the key points), semantic-based key phrase extraction performs robustly and the best. Statistics-based and graph-based approaches are relatively sensitive to the length of the model answer. As more description is provided, more key phrases will be extracted. Thus, the learner answer gets a higher chance to be successfully matched to the model answer. In summary, we could leverage semantic-based segmentation and simplify the model answer design with a list of key points. It means we can reduce the human efforts in designing comprehensive grading rubrics if that does not bring additional performance improvement.

Experiment Five. CRQ Two with key phrase extraction and semantic embedding In this experiment, we apply the same approach as in Experiments One to Four to CRQ Two to evaluate whether the insights derived are generalizable to other CRQs. We apply the same set of key phrase extraction methods to both learner answer and model answer. We use contextualized embedding for text representation with BERT basic and BERT fine-tuned LMs (that is proved to be the most effective approach from Experiments One to Four). We set semantic similarity thresholding as 88% (proved to be the best from Experiment 3). We set

Accuracy	POS		TFIDF		RAKE		TextRank		BERT basic		BERT finetuned	
	TPR	FPR	TPR	FPR	TPR	FPR	TPR	FPR	TPR	FPR	TPR	FPR
85%*	0.24	0.25	0.89	0.47	0.84	0.41	**0.72**	**0.21**	**0.75**	**0.26**	**0.75**	**0.25**
88%*	0.12	0.10	**0.73**	**0.25**	**0.69**	**0.22**	0.68	0.25	0.60	0.15	0.60	0.16
85%**	0.69	0.33	0.90	0.49	0.91	0.50	**0.83**	**0.31**	0.84	0.35	0.84	0.35
88%**	0.46	0.19	**0.78**	**0.30**	**0.79**	**0.31**	0.68	0.25	**0.74**	**0.22**	**0.75**	**0.22**

Table 3: Model answer segmentation comparison. We apply BERT fine-tuned segmentation for learner answers. We use distilbert-base-nli-mean-tokens as the LM for semantic embedding. We set hyperparameters top key phrase percentage for all the methods as 100%. * For model answer only. ** For model answer with grading rubrics. Model answer is less sensitive to the segmentation approaches. Grading rubrics help improve performance for non-semantic-based segmentation.

	POS			TFIDF			RAKE			TextRank			BERT basic		
	a	b	c	a	b	c	a	b	c	a	b	c	a	b	c
1	0.53	0.57	0.54	0.64	0.64	0.57	0.61	0.60	0.54	0.46	0.67	0.46	0.67	0.67	0.46
2	0.57	0.57	0.54	0.60	**0.71**	0.60	0.57	0.67	0.68	0.64	0.67	0.61	0.64	**0.71**	0.58
3	0.57	0.64	0.54	0.60	0.64	0.5	0.60	0.64	0.5	0.61	0.50	0.54	**0.71**	**0.71**	0.54
4	0.57	0.64	0.46	0.60	**0.71**	0.58	0.50	**0.71**	0.58	0.50	0.64	0.54	0.67	**0.78**	0.54
5	0.5	0.60	0.46	0.60	**0.71**	0.5	0.60	**0.71**	0.5	**0.78**	**0.71**	0.68	**0.71**	**0.78**	0.54

Table 4: Experimental results for CRQ2 with contextualized embedding. The metric is accuracy. 1: POS, 2: TFIDF, 3: RAKE, 4: TextRank, 5: BERT basic for key phrase extraction for learner answer. *a* refers to BERT-base-cased. *b* refers to distilbert-base-nli-mean-tokens. *c* refers to en-core-web-sm. BERT-based segmentation outperforms for both learner answer and model answer.

the top key phrase percentage as 100% as a default practice (statistics-based and graph-based approaches can be optimized further). For the spacy model, we set the top key phrase percentage as 70% as a default practice. Because there is only one key point in the model answer, we report the accuracy ((TP + TN)/Total) only for comparison. As Table 4 shows, BERT basic key phrase extraction outperforms others in both learner answer segmentation and model answer segmentation. The model answer with the BERT basic approach achieves 71%, 78% accuracy with learner answer segmented with TFIDF, RAKE, TextRank, and BERT basic. The learner answer with the BERT basic approach also achieves 71%, 78% accuracy with model answer segmented with TFIDF, RAKE, TextRank, BERT basic. BERT basic on both learner answer and model answer achieves the best performance, 78%. BERT2 (fine-tuned) LM performs better than BERT1 (BERT basic) LM. All being said, semantic-based key phrase extraction outperforms non-semantic-based approaches. Fine-tuning LM on a similar task could further improve the performance by better contextualizing the text.

Insights summarization From our experiments, we summarize the following insights. For both learner answer and model answer segmentation, semantic-based key phrase extraction outperforms statistics-based and graph-based methods, especially on FPR measure. Simply put, understanding the semantics plays a significant role in accurately extracting the key phrases. Both statistics-based and graph-based methods can be optimized by fine-tuning hyperparameters (top key phrase percentage and semantic embedding similarity thresholding) to deliver a decent solution. Model answer segmentation is less sensitive to segmentation approaches compared to learner answer segmentation. It indicates that well-written text (model answer) could make key phrase extraction easier even without understanding the semantics. For semantic embedding, contextualized embedding achieves better results compared to traditional statistics-based (word frequency, co-occurrence matrix) embedding. Simply put, contextualizing the semantics in representing the text could deliver a more accurate result. Traditional statistics-based embedding could also deliver a decent solution with hyperparameter fine-tuning. Fine-tuning an LM with a similar task could also improve the performance by better contextualizing the text representation. Further, semantic-based key phrase extraction methods are robust to the length of the model answer as long as the semantics at key point level is provided. In other words, the efforts of manually extracting rubrics (from learner answers) that conveys similar information could be saved. Meanwhile, for statistics-based and graph-based key phrase extraction, detailed grading rubrics could help improve the model performance with certain level of parameter optimization.

Conclusions and future work

In this work, we introduce an approach to generate targeted feedback for a learner answer by leveraging model answers and state-of-the-art NLP techniques. The proposed approach includes key phrase extraction and semantic embedding for analyzing the misconceptions. We test three classes of key phrase segmentation methods and two classes of semantic embedding methods. From experiments, we demonstrate that semantic-based segmentation outperform statistics-based and graph-based methods. We conclude that contextualized semantic embedding with fine-tuning on similar tasks performs the best in capturing the semantics. We also leverage learner answers to design grading rubrics and test incorporating the grading rubrics into model answers. The result shows a potential opportunity to reduce human efforts in designing comprehensive grading rubrics when we use semantic-based segmentation and embedding techniques.

This work sets a baseline for misconception analysis for a CRQ activity. We would continue this work with building a conversational agent to nudge learners with feedback to help them expand on their answers and resubmit. Specifically, we would research refining course design by understanding how learners use CRQ activities in their learning. We would also work on providing personalized feedback considering learners' knowledge state by incorporating other evidences collected through the learning platform. Also, we would like to get feedback from learners on the feedback they received on their answer.

The proposed approach is tested and implemented on English corpus and for conceptual knowledge in an open-navigation online learning system. Currently, the approach leverages the existing LMs to understand the learner answer and model answer. It would be interesting to test whether the approach is applicable or extendable to other languages or in other formats of context, such as code or equations.

References

Arter, J., and McTighe, J. 2001. *Scoring rubrics in the classroom: Using performance criteria for assessing and improving student performance*. Corwin press.

Basu, S.; Jacobs, C.; and Vanderwende, L. 2013. Powergrading: a clustering approach to amplify human effort for short answer grading. *Transactions of the Association for Computational Linguistics* 1:391–402.

Blaž Škrlj, Andraž Repar, S. P. 2019. Rakun: Rank-based keyword extraction via unsupervised learning and meta vertex aggregation. *Statistical Language and Speech Processing 2019*.

Boudin, F. 2018. Unsupervised keyphrase extraction with multipartite graphs. *In Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT, New Or- leans, Louisiana, USA, June 1-6, 2018, Volume 2*.

Butler, A. C.; Godbole, N.; and Marsh, E. J. 2013. Explanation feedback is better than correct answer feedback for promoting transfer of learning. *Journal of Educational Psychology* 105(2):290.

Campos R., Mangaravite V., P. A. J. A. N. C. J. A. 2018. A text feature based automatic keyword extraction method for single documents. *ECIR 2018*.

Carbonell, J., and Goldstein, J. 1998. The use of mmr, diversity-based reranking for reordering documents and producing summaries. In *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, 335–336.

Clark, R. C., and Mayer, R. E. 2016. *E-learning and the science of instruction: Proven guidelines for consumers and designers of multimedia learning*. John Wiley & Sons.

Frost, E. R. 2016. Feedback distortion: The shortcomings of model answers as formative feedback. *Journal of Legal Education* 65(4):938–965.

Huxham, M. 2007. Fast and effective feedback: are model answers the answer? *Assessment & Evaluation in Higher Education* 32(6):601–611.

Jeffrey Pennington, Richard Socher, C. M. 2014. Glove: Global vectors for word representation. *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Lan, A. S.; Vats, D.; Waters, A. E.; and Baraniuk, R. G. 2015. Mathematical language processing: Automatic grading and feedback for open response mathematical questions. In *Proceedings of the Second (2015) ACM Conference on Learning@ Scale*, 167–176.

Leacock, C., and Chodorow, M. 2003. C-rater: Automated scoring of short-answer questions. *Computers and the Humanities* 37(4):389–405.

Mihalcea, R., and Tarau, P. 2004. Textrank: Bringing order into text. in proceedings of the 2004 conference on empirical methods in natural language processing. *EMNLP 2004, Barcelona, Spain, July 25-26, 2004, A meeting of SIGDAT, a Special Interest Group of the ACL, held in conjunction with ACL 2004*.

Mihalcea, R., and Tarau, P. 2008. Single document keyphrase extraction using neighborhood knowledge. *AAAI 2008, Chicago, Illinois, USA, July 13-17, 2008, 855–860*.

Mohler, M., and Mihalcea, R. 2009. Text-to-text semantic similarity for automatic short answer grading. In *Proceedings of the 12th Conference of the European Chapter of the ACL (EACL 2009)*, 567–575.

Mohler, M.; Bunescu, R.; and Mihalcea, R. 2011. Learning to grade short answer questions using semantic similarity measures and dependency graph alignments. In *Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies*, 752–762.

Reimers, N., and Gurevych, I. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*.

Reimers, N. 2020. sbert.

Rose, S., E. D. C. N., and Cowley, W. 2010. Automatic keyword extraction from individual documents. *Text Mining: Applications and Theory, 1–20*.

Ruder, S. 2017. An overview of multi-task learning in deep neural networks. *arXiv preprint arXiv:1706.05098*.

Saha, S.; Dhamecha, T. I.; Marvaniya, S.; Sindhgatta, R.; and Sengupta, B. 2018. Sentence level or token level features for automatic short answer grading?: Use both. In *International conference on artificial intelligence in education*, 503–517. Springer.

Selvi, P., and Bnerjee, A. 2010. Automatic short-answer grading system (asags). *arXiv preprint arXiv:1011.1742*.

Sultan, M. A.; Salazar, C.; and Sumner, T. 2016. Fast and easy short answer grading with high accuracy. In *Proceedings of the 2016 Conference of the North American Chapter* *of the Association for Computational Linguistics: Human Language Technologies*, 1070–1075.

Süzen, N.; Gorban, A. N.; Levesley, J.; and Mirkes, E. M. 2020. Automatic short answer grading and feedback using text mining methods. *Procedia Computer Science* 169:726–743.

Wiggins, G.; Wiggins, G. P.; and McTighe, J. 2005. *Understanding by design*. Ascd.