A Machine Learning Approach for Suggesting Feedback in Textual Exercises in Large Courses

Jan Philip Bernius, Stephan Krusche, and Bernd Bruegge
Department of Informatics
Technical University of Munich
Munich, Germany
janphilip.bernius@tum.de, krusche@in.tum.de, bernd.bruegge@tum.de

ABSTRACT

Open-ended textual exercises facilitate the comprehension of problem-solving skills. Students can learn from their mistakes when teachers provide individual feedback. However, courses with hundreds of students cause a heavy workload for teachers: providing individual feedback is mostly a manual, repetitive, and time-consuming activity.

This paper presents **CoFee**, a machine learning approach designed to suggest computer-aided feedback in open-ended textual exercises. The approach uses topic modeling to split student answers into text segments and language embeddings to transform these segments. It then applies clustering to group the text segments by similarity so that the same feedback can be applied to all segments within the same cluster.

We implemented this approach in a reference implementation called **Athene** and integrated it into Artemis. We used Athene to review 17 textual exercises in two large courses at the Technical University of Munich with 2,300 registered students and 53 teachers. On average, Athene suggested feedback for 26% of the submissions. Accordingly, 85% of these suggestions were accepted by the teachers, 5% were extended with a comment and then accepted, and 10% were changed.

Author Keywords

Software Engineering, Education, Interactive Learning, Automatic Assessment, Grading, Assessment Support System, Learning, Feedback

CCS Concepts

•**Social and professional topics** → **Software engineering education;** •**Computing methodologies** → *Natural language processing;*

INTRODUCTION

The rise in student numbers in universities has led to an increase in course management efforts, and made it challenging to provide high-quality individual feedback to students [19]. Recent approaches, such as online platforms and live streaming, allow teachers[1] to cope and interact with a large amount of students on an individual level, regardless of the respective course size.

In particular, large university courses with hundreds of students rely on teaching assistants to provide feedback on exercises, e.g., multiple-choice quizzes and textual exercises. Multiple-choice quizzes are easy to assess, and tools are broadly available in learning management systems (LMSs) and for paper-based assessment. However, mastery of these quizzes does not require problem-solving skills because they typically target only lower cognitive skills, in particular, *knowledge* recall and *comprehension*. Most quiz types include predefined options and do not reflect work practices in industry. It is difficult to create quizzes that stimulate higher cognitive skills, such as problem-solving, which are important in computer science [1, 32].

Open-ended textual exercises allow instructors to teach problem-solving skills and allow students to improve their knowledge. These exercises do not have a single correct solution, but rather allow answers within a particular solution space which can be characterized by words and phrases. The searchlight theory of scientific knowledge [27] states that students increase their knowledge through observations, especially observations that prove their assumptions wrong. Students profit from having an individual feedback relationship with their teachers [10]. Individual feedback and formative assessments are essential elements in learning [11, 12]. Feedback on open-ended exercises allows students to try out problem-solving and to experience failure. Students need guidance in the form of feedback in their learning activities to prevent misconceptions [13].

[1]For this paper, we define teachers as both instructors and teaching assistants (see Figure 1). Instructors are employees of the university such as professors, lecturers, and doctoral candidates. Teaching assistants are experienced students who have passed the same course previously with a good grade and who are motivated to help in the teaching process. Some universities also use the term "tutor" to refer to a teaching assistant.

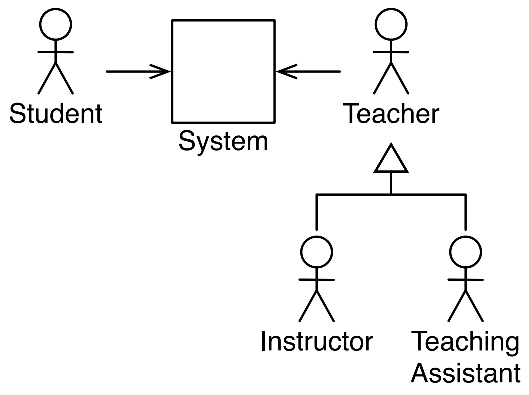


Figure 1. Use case diagram of the Athene and Artemis system. Students and teachers interact with the system. Teachers are instructors, employees of the university, or teaching assistants, who are previous students hired to assist in teaching.

However, textual exercises lead to greater variability because students need to formulate individual answers to problems. This results in high manual effort when reviewing students' answers. Assuring consistent feedback is difficult with a large number of teaching assistants. In this paper, we present an approach for computer-aided feedback for textual exercises that addresses these challenges. We implemented the approach in an open-source reference implementation, used it in multiple exercises, and evaluated its effects on the learning experience of students. In particular, we investigated the following research questions (RQ):

RQ1 **Coverage:** How much feedback can be automatically suggested?

RQ2 **Accuracy:** How accurate is the suggested feedback?

RQ3 **Quality:** How do students perceive the quality of the automatically suggested feedback?

The paper is organized as follows: Section 2 describes the background of this work which consists of machine learning concepts, in particular language models. In Section 3, we show similar approaches and relate them to the approach presented in this paper. Section 4 presents the approach computer-aided feedback for textual exercises (CoFee) based on supervised learning to deal with the greater variability in the student answers. Language embeddings and clustering are used to provide individual feedback based on similarity. Section 5 shows the open-source reference implementation **Athene**[2] which is integrated into the open-source LMS Artemis[3]. Section 6 describes the courses in which the approach was used, shows the study design of the empirical evaluation with respect to Bloom's revised taxonomy [2], presents results and limitations, and discusses the findings. Section 7 concludes the paper with its main contributions and future work.

BACKGROUND: LANGUAGE MODELS

Assessing text submissions automatically requires comparing segments of those submissions and identifying similar pieces of text. Therefore, we need a measurable abstraction of a texts meaning as an intermediate representation. This paper relies on existing approaches and techniques from the domain of natural language processing (NLP), most notably language models and word embeddings, to convert a piece of text into a comparable format. Student answers can contain unknown words, incorrect use of grammar and punctuation, and false statements.

Word embedding is a feature learning technique in NLP, where words or phrases from the vocabulary are mapped to vectors of real numbers (each word is associated with a point in a vector space) [21]. The feature vector represents different aspects of the word and consequently, words that have the same meaning are assigned similar vector representations. Additionally, word embeddings are capable of capturing word analogies by examining various dimensions of the differences between word vectors [24]. For example, the analogy "king is to queen as man is to woman" should be encoded in the vector space by the vector equation $king - queen = man - woman$.

The distributed representation is learned based on the usage of the words. This allows words that are used in similar contexts to have similar representations, naturally capturing their meaning. ELMo [26] is a word embedding constructed as a task-specific combination of the intermediate layer representations in a bidirectional language model (biLM). It models complex characteristics of words-use in the language dictated by the syntax and semantics. It also captures how these uses vary across linguistic contexts, which is important for addressing polysemy in natural languages.

In a deep language model (LM), the higher-level long short term memory (LSTM) states are shown to capture context-dependent aspects of word meaning while lower-level states model aspects of the syntax. By constructing a representation out of all the layers of the LM, ELMo is able to capture both characteristics of the language. ELMo representations have three main characteristics that allow them to achieve state-of-the-art results in most common NLP downstream tasks. First, ELMo representations are contextual: the representation for each word depends on the entire context in which it is used. They are also deep: the word representations combine all layers of a deep, pre-trained language model neural network. Finally, ELMo representations are purely character based, allowing the network to use morphological clues to form robust representations for out-of-vocabulary tokens, unseen in training.

RELATED WORK

Automated essay scoring (AES) computes scores on written solutions based on previous submissions. AES systems require a perfect solution to be available up front [23, 31]. They primarily consider the distance to a perfect solution to determine the grade. Feedback is not the focus. Manual clustering and shared grading are concepts used in research [25] and commercial tools (i.e., Gradescope). Managing clusters is hard at scale, communicating the exact differences between clusters between many graders.

[2]Athene: https://github.com/ls1intum/Athene
[3]Artemis: https://github.com/ls1intum/Artemis

Atenea is a computer-assisted assessment system for scoring short answers in computer science [25] and is integrated into a web-based application. Atenea uses a database of questions with a correct sample solution for each, either written by a teacher or taken from a highly graded student's answer. When a student accesses Atenea, a random question from this pool is asked and compared to the given sample solution by utilizing a hybrid for syntax as well as semantic similarity. The system works by combining latent semantic analysis (LSA) and a modified bilingual evaluation understudy (BLEU) algorithm, with the hypothesis that syntax and semantics complement each other naturally. The combination of both NLP tools always performs better (with a higher hit rate) than their individual parts, with the authors believing that combinations of syntactical and semantical analysis can lead to even greater results for automatic text assessment.

Atenea compares student answers to a set of predefined answers. The grade is determined by its similarity to these predefined answers. This approach is limited to exercises with a narrow answer space where possible answers are known beforehand. A high variability in answers requires a large set of predefined answers, which limits the applicability of the system. The focus of the Atenea system is grading, whereas Athene is primarily focused on individual feedback. Athene does not require a predefined solution but collects knowledge on correct and incorrect solutions during the manual assessment. The evaluation of the Atenea authors focuses on a comparison of NLP techniques in the grading context and is based on a dataset. We evaluate Athene by using it in multiple courses and measuring its performance.

Powergrading is an automatic assessment approach [3]. Instead of solely focusing on providing a numerical score or a right or wrong grade, Powergrading tries to justify a certain given grade by providing feedback in the form of a comment as to why an answer is right or wrong, similar to how a teacher would do it in a classroom setting. Basu et al. propose a system, that clusters similar answers to a question so that teachers can "divide and conquer" the correction process by assigning a whole cluster with the same score and comment, therefore reducing the correction time significantly. Clustering answers to a question should happen based on a distance function, which is composed of different features and tries to learn a similarity metric between two students' answers automatically. Some of the implemented and used features that are weighted in developing this distance function used for clustering are, e.g., the difference in length between two answers, the term frequency-inverse document frequency (TF-IDF)[4] similarity of words, or the LSA vectorial score based on the entirety of Wikipedia as a training text corpus. The authors have tested their implementation with test data from the United States Citizenship Exam in 2012 with 697 examinees and concluded that around 97% of all submissions can be grouped into similar clusters so that teachers would only have to provide feedback for a single cluster and would still be able to reach and correct multiple submissions at once, therefore reducing assessment time significantly [3].

[4]TF-IDF: An information extraction statistic which indicates how significant a word is to a document [28].

Powergrading is focused on short-answer grading, where a typical answer does not exceed two sentences. Athene is not limited to a certain answer length and uses segmentation to work with multiple sentences or paragraphs. Similar to Powergrading, Athene groups segments into clusters. Both systems assume hierarchical cluster structures. Powergrading allows teachers to grade clusters rather than submissions, whereas Athene will use the cluster structure to suggest feedback for following assessments.

Gradescope is a system geared toward the assessment of handwritten homework and exam exercises [30] by scanning paper-based work. Teachers review the submissions online. Gradescope allows the teacher to dynamically create grading rubrics at the assessment time. For the assessment, teachers can group similar submissions manually for shared grading or relay on suggested groups.[5]

Athene follows a similar idea by sharing feedback with groups of answers; however, Athene groups individual segments, whereas Gradescope groups entire submissions. Gradescope allows the grader to grade multiple submissions as one, similar to Powergrading, whereas Athene shares individual feedback elements across multiple submissions. Athene requires teachers to inspect every submission and supports by suggesting feedback items. Neither system requires a training dataset of previously assessed answers. For exercises with a limited answer spectrum, Gradescope does allow the grader to assess several submissions efficiently as it reduces the number of solutions to grade. However, for exercises with high variability in answers (e.g., when asking for examples), this approach is more limited as more groups with less elements need to be reviewed.

APPROACH: COMPUTER-AIDED FEEDBACK (COFEE)

CoFee uses supervised machine learning to learn correct answers and related feedback. Figure 2 shows the main workflow how CoFee can automatically propose computer-aided feedback to students' answers. CoFee learns which answers to an exercise are correct and which are incorrect. For further submissions, the learning platform can automatically generate suggestions for similar answers or even evaluate the answers fully automatically. In doing so, the learning platform uses the knowledge of previous assessments by lecturers. The more students participate in an exercise, the more knowledge is generated and the better feedback the learning platform can suggest.

Figure 3 shows the details of the activity "preprocess answers" shown in Figure 2 and represents the basis for the three objectives mentioned above. The system analyzes incoming text (responses) using NLP, divides them into text segments, and uses them to create text clusters with similar text segments from different responses. This is done using a combination of segmentations and linguistic embeddings, in particular deeply contextualized word representations (ELMo). This allows for an understanding students' responses and for the generation of individualized feedback. In this way, a learning platform can automatically reuse manual feedback for contributions

[5]https://gradescope.com

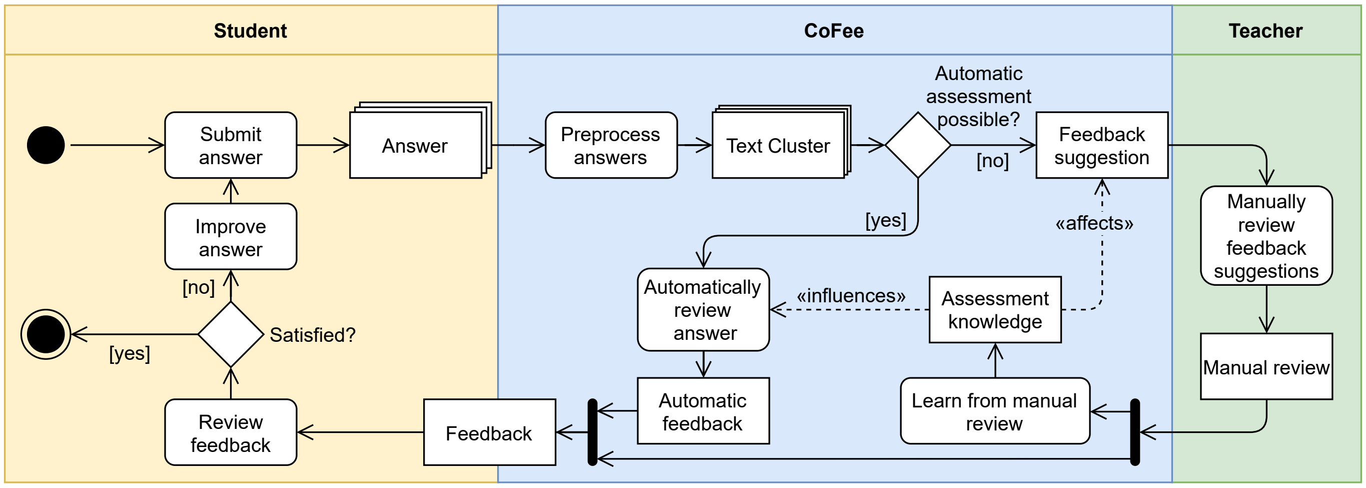


Figure 2. Workflow of automatic assessment of submissions to textual exercises based on the manual feedback of teachers. CoFee analyzes manual assessments and generates knowledge for the suggestion of computer-aided (automatic) feedback (UML activity diagram).

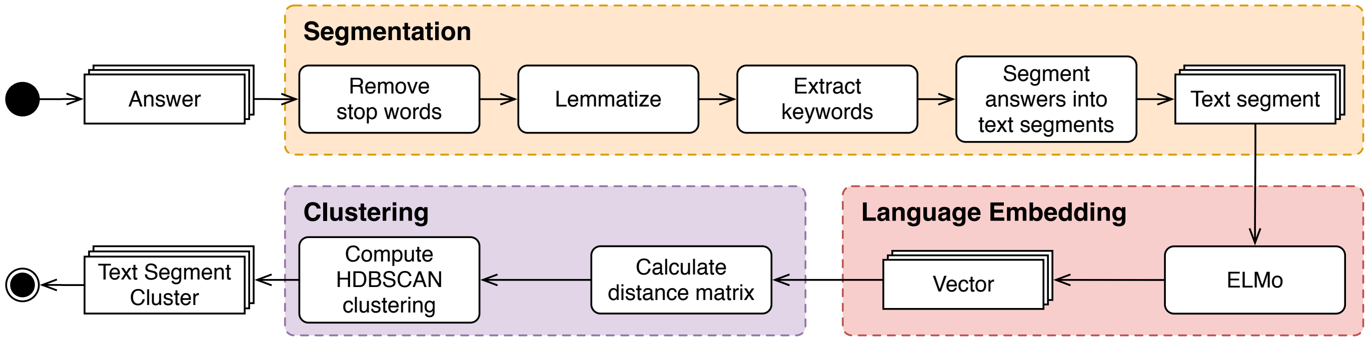


Figure 3. Detailed overview of the machine learning activities as part of the "preprocess answers" activity in Figure 2. These are used to extract text segments and build text clusters for scoring and similarity analysis (UML activity diagram).

from different students. This can reduce the workload for teachers and increase the consistency and quality of feedback to improve students' understanding.

The goal is to increase the quality and quantity of the feedback provided to students while decreasing the overall assessment time. CoFee integrates into existing learning platforms that need to provide an interface for students to submit their textual answers. We utilize a segment-based feedback concept [5], requiring assessors to provide feedback and score in relation to a segment of student's answer, resulting in relatable and reusable feedback elements.

CoFee trains its assessment model with every feedback element and thereby becomes more accurate with every new feedback element. After the assessment process, the system can detect conflicting assessments in both comments and scores. Therefore, CoFee computes the similarity among feedback comments. We claim that the distance between two text segments should be proportional to the distance between the feedback comments. If this relation is violated, CoFee prompts the teacher to review the pair of submissions and allows them to update the assessment as needed. The learning platform may only release the feedback to students after the teachers have the chance to resolve inconsistencies.

Compared to existing work, our system segments and clusters student solutions automatically. By training the system during the assessment process, we do away with the need for a reference dataset before the assessment. Furthermore, by training with highly and lowly scored solutions, we maintain a dataset to provide helpful feedback comments to support the learning process. Dynamically collecting the dataset during assessment keeps the system independent of any domain and allows for use of the system with new exercises to incorporate the latest knowledge into teaching.

REFERENCE IMPLEMENTATION (ATHENE)

We implemented CoFee in a reference implementation called Athene [4] that is integrated into the learning platform Artemis [15]. After the exercise deadline, Artemis sends the students' answers to Athene for processing. Athene will preprocess the answers before the assessment begins and will identify segments suitable for the same feedback. Figure 3 depicts the preprocessing activities. This represents the basis for the three objectives mentioned above. The system analyzes incoming student answers using NLP, divides them into text segments, and uses them to create text clusters with similar text segments from different responses. Figure 4 depicts the top-level design of the system which consists of three steps: segmentation, language embedding and clustering.

First, Athene analyzes the answers to identify segments [6, 7]. Therefore, Athene identifies common topics described in the answers from all students. A topic is represented by a keyword. To identify the important topics for an exercise, Athene counts the occurrences of lemmatized words across all students and selects the 10 most common words [7]. Within each student answer, Athene will break down all submissions into clauses. Adjacent clauses that share the same topic, represented by the use of a keyword and absence of a new keyword, are merged to form a segment. If a new keyword appears an a following clause, we identify a topic shift and start a new segment. This results in a set of topically coherent segments.

Second, Athene uses an ELMo model to convert each segment to vector form. ELMo vectors have 1,024 dimensions representing the information extracted from the segment. The vector representation allows for a comparison of segments and for the identification of similarities. Athene uses a pre-trained ELMo model [26] based on a dataset consisting of 5.5B tokens from Wikipedia and news articles.[6]

Third, Athene employs the Hierarchical Density-Based Spatial Clustering (HDBSCAN) clustering algorithm [22] to identify classes of similar text segments. Within a cluster, Athene shares manually created feedback as suggestions. The hierarchical clustering algorithm allows for a determination of the required number of clusters dynamically. Further, the hierarchical structure is used to dynamically narrow or widen the search radius depending on the availability of feedback. Narrow clusters provide more accurate feedback on the one side; however, they also limit the possible coverage. Larger clusters increase the possibility to find existing feedback to compose a suggestion; however, they also increase the risk of false feedback.

During the manual assessment, Athene sorts submissions so that it priorities submissions with the highest effect on automated grading. Submissions with several segments in clusters without feedback are prioritized, maximizing the possible coverage for automatic feedback suggestions. For each segment, Athene searches their respective clusters for existing feedback and suggests the closest feedback. Furthermore, credit points associated with feedback are used to prioritize based on the clusters' credit average. Athene's automatic feedback suggestions are displayed to teachers within Artemis as part of the review interface [5], as depicted in Figure 5. Teachers can add additional feedback to unreviewed parts of the student solution. They can either approve of the feedback suggestions or update them as they see fit.

EVALUATION

After several teachers used Athene in initial experiments in smaller courses with around 500 students, they found anecdotal evidence that the system improves the quantity and quality of feedback. The next step was to evaluate the approach in multiple exercises in the course *Introduction to Software Engineering (SE1)* with 1,800 students and 49 teaching assistants and in a second course *Networks for Monetary Transactions*. In this section, we describe the two courses and the study

[6]AllenNLP – ELMo: https://allennlp.org/elmo

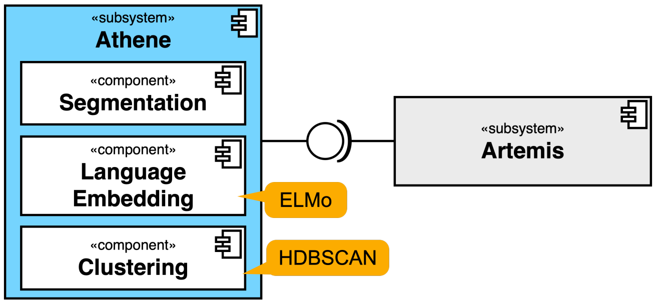
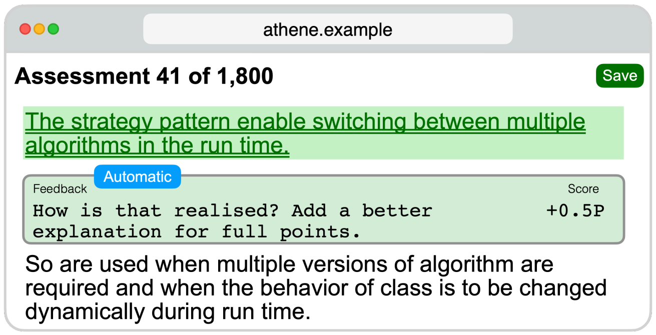


Figure 4. Top-level design of Athene, which is decomposed into three subsystems for segmentation, language embedding, and clustering and offers an API to be used in existing LMS (UML component diagram).



Assessment 41 of 1,800

The strategy pattern enable switching between multiple algorithms in the run time.

```
How is that realised? Add a better        +0.5P
explanation for full points.
```

So are used when multiple versions of algorithm are required and when the behavior of class is to be changed dynamically during run time.

Figure 5. Example of the teacher interface: Athene presents a feedback suggestion for the first text segment with a feedback comment and a score.

design of the evaluation. We show the results of the usage of Athene and discuss the findings, implications, and limitations.

Courses

The course SE1 is an introductory software engineering course, with around 1,800 registered students who are mainly computer science bachelor's students in their second semester. Students with computer science as a minor can also enroll in the course. The course covers software engineering concepts, such as requirements analysis, system and object design, testing, lifecycles, configuration management, and project management and covers UML modeling [19]. To participate in the course, students need to have fundamental programming experience (e.g., CS1). The instructors use constructive alignment [8] to align the teaching concepts and exercises with the course objectives. For each lecture, they define learning goals based on six cognitive processes in Bloom's revised taxonomy [2]. The course focuses on higher cognitive processes: students apply the concepts in concrete exercises.

Following an interactive learning approach, SE1 teaches software engineering concepts with multiple, small iterations of theory, example, exercise, solution and reflection [16]. It utilizes exercises to foster student participation [17] and to motivate the students to attend the lectures [18]. The course involves different kinds of exercises:

1. Lecture exercises as part of the (virtual) lectures
2. Group exercises solved in small ad hoc groups
3. **Homework exercises** to be solved throughout the week individually

4. Team exercises to be solved in a team in five 2-week periods
5. **Exam exercises** to assess the students' knowledge after the course has finished in multiple variants

Students were asked to submit their solutions to all exercises but group exercises to Artemis to receive an assessment with feedback and points. The students could gain bonus points for the final exam when participating in the exercises. To train software engineering and problem-solving skills, the instructors utilize programming, modeling, **textual**, and quiz exercises in the course. Automatic assessment suggestions based on Athene have been enabled for 11 textual homework exercises and six textual exam exercises.

The course *Networks for Monetary Transactions* has the learning goals to understand and assess the fundamentals, architecture, and security of domestic and international payment networks and their legal frameworks. Around 500 students participated. The teachers used Artemis to conduct an online exam during the COVID-19 pandemic. The exam consisted of 11 quiz exercises and three text exercises. Automatic assessment suggestions based on Athene were enabled for one **textual** exam exercise: *IT-Attacks*.

Bloom created the taxonomy of educational objectives, defining six categories: Knowledge, Comprehension, Application, Analysis, Synthesis, and Evaluation [9]. The *revised taxonomy* shifts the focus from static educational objectives toward a classification of cognitive processes students encounter when solving exercises [2]. The exercises conducted as part of the evaluation can be classified to train the cognitive processes *Remember*, *Understand*, but *Apply*, and *Analyze* (see Figure 6).

Table 1 lists the textual exercises and includes the cognitive process (as a category) that receives the most training in terms of the revised taxonomy. Some exercises such as *H09E02* and *H10E01* facilitate understanding by asking student to explain concepts. *H10E01*, e.g., states: "Name and explain similarities and differences between the Unified Process and Scrum in your own words". Other exercises such as *Exam 3* focus on the application of knowledge. Students need to apply their requirements elicitation skills in order to create use case descriptions based on a given problem statement.

Study Design

Figure 7 shows the study design of the evaluation that was instantiated for each exercise in which Athene was used for grading. The teacher defines the exercise in Artemis with a problem statement, grading criteria, example solutions, and a due date. The students can insert their solution in plain text on Artemis. After the due date, Artemis sends all student answers to Athene to preprocess the answers as described in the Approach section. The teachers can start reviewing the student answers as soon as Athene completes the preparation and stores the text clusters. For every student answer, the teachers create a review consisting of multiple feedback items. During the review phase, the teachers used a chat room to discuss the grading criteria as needed.

Every review can either be computer-aided, if at least one feedback item is suggested by the system, or manual. Furthermore,

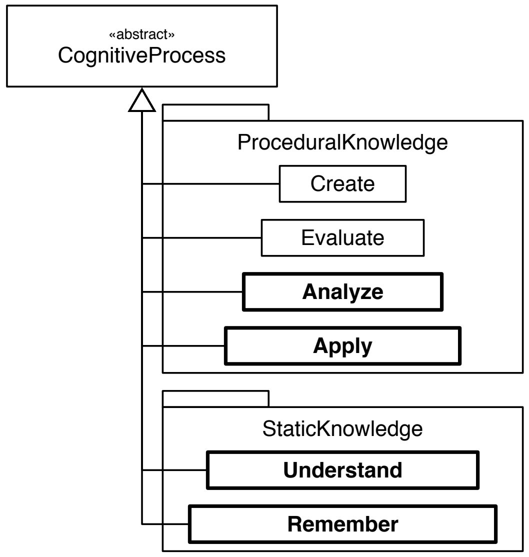


Figure 6. Exercises in the evaluation assess different cognitive processes. This taxonomy, based on the revised Bloom's taxonomy [2], depicts the hierarchy of skills. Exercises test static knowledge by testing the *remember* and *understand* skills but also *apply* and *analyze*, e.g., by identifying design issues from a system.

Athene stores intermediate versions of all feedback items to evaluate how teachers work with feedback suggestions.

After the teachers completed the review, we retrieved the classification of the reviews from the Artemis database using SQL queries. Two researchers verified the correctness of the queries. We collected the statistics on the feedback items from Athene. We inserted the measurements in a spreadsheet for further analysis and graphing. Two researchers reviewed the results for consistency and plausibility and took several samples to check individual feedback entries.

Results

The presentation of the results is based on the research questions stated at the beginning of the paper on the coverage, accuracy and quality of the approach.

Review Coverage

First, we classify the reviews into two categories: manual and computer-aided. A review is considered computer-aided, if at least one feedback item was suggested by Athene. Figure 8 depicts the classification of the reviews. On average, 26% of all reviews were computer-aided. The system performed best in exercise `Exam 1`, with 70% computer-aided reviews. Exercises `Exam 4` and `Exam 6` have the least coverage, with 2% and 8% computer-aided reviews, respectively. However, Athene was disabled for exercise `Exam 4` after a few assessments.

Finding 1: Coverage: Athene can cover up to 70% of reviews with feedback suggestions without previous training data or a predefined solution.

Exercise	Title	Category
H04E01	Coupling and Cohesion	Understand
H04E02	Analysis Models & System Design	Analyze
H04E03	Design Goal Trade-offs	Apply
H05E02	Centralized vs Decentralized Designs	Understand
H06E03	Specification & Implementation Inheritance	Apply
H06E04	Inheritance vs. Delegation	Understand
H07E03	MVC & Observer Pattern	Understand
H09E01	Advantages and Disadvantages of Scrum	Understand
H09E02	Unified Process and Scrum	Understand
H10E01	Problems using Git	Understand
H10E02	Merge Conflicts & Best Practices	Understand
Exam 1	Requirements	Apply
Exam 2	Visionary Scenarios	Apply
Exam 3	Use Cases	Apply
Exam 4	Access Control	Apply
Exam 5	Design Goal Trade-offs	Apply
Exam 6	Centralized vs. Decentralized Control	Apply
Exam 7	IT-Attacks	Remember

Table 1. Homework and exam textual exercises and their categorization following Bloom's revised taxonomy [2] used in the evaluation.

Feedback Accuracy

Second, we classified feedback items based on the intermediate versions collected during the review process. Feedback items can be classified as follows:

1. A feedback suggestion that remains unchanged is classified as *automatic*.
2. For changed suggestions, Athene computes the Levenshtein distance [20] between feedback comments. Athene classifies a changed feedback as a *typo* fix for a Levenshtein distance > 0.9.
3. Athene uses the longest common substring length and the Jaro–Winkler distance [33] to recognize feedback suggestions with a manual *extended* comment.
4. Feedback not classified in these metrics is considered as *changed*.

We analyzed the teachers' assessment work for two homework and seven exam exercises. The results depicted in Figure 9 show that on average, 85% of computer-aided feedback comment suggestions remained unchanged in their final assessment or only had minor modifications, such as corrections to typing mistakes. Furthermore, 5% of suggested comments were extended with additional feedback at the end of the suggestion to provide more details for the student. The remaining 10% of comments were changed. In these cases, the comment was either rewritten from scratch or was heavily revised.

Finding 2: Accuracy: On average, 85% of the feedback suggestions are accurate and can be published to students without modification.

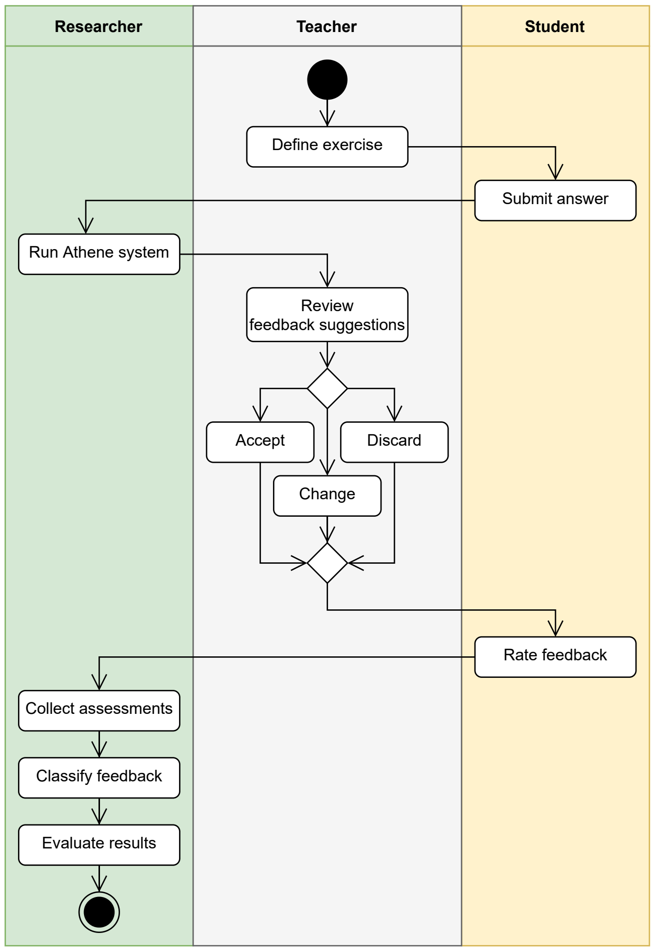


Figure 7. Research approach depicted with the involved actors and flow of events (UML activity diagram).

Perceived Quality

Third, we asked students to rate their received feedback on a 5-star scale. The students rated a total of 396 reviews out of $10,240$ total reviews done by the teachers. Artemis presents the rating input underneath the feedback and asks, *"How useful is the feedback for you?"* Figure 10 depicts the distribution by star rating. In the study, 85% of the ratings were either 1-star or 5-star ratings. Students with computer-aided feedback were more likely to give a 5-star rating (72%) when compared to students who received manual feedback (62%). On the same page, computer-aided feedback received 1-star ratings less often (14%) than manual feedback (25%). Students giving a 5-star rating on average (92% and 89%, respectively) had better scores than students giving 1-star ratings (69% and 66%, respectively).

Finding 3: Quality: The computer-aided feedback in Athene has at least the same quality as manual feedback.

Limitations

This section discusses threats to the trustworthiness of the presented results, and whether the results are biased based on the researchers' subjective point of view. We distinguish

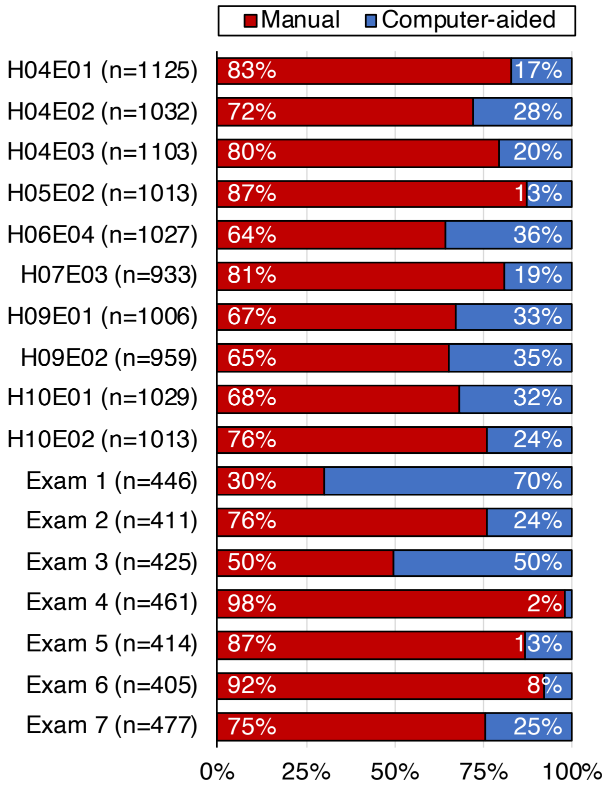


Figure 8. Exercises with their assessment ratios. Computer-aided reviews received automated grading suggestions which were reviewed by a teacher. On average, 26% reviews were computer-aided.

between three aspects of validity: internal validity, external validity, and construct validity [29].

Internal Validity: The accuracy of the feedback suggestions is measured by the acceptance of the teacher. A second review from a control teacher would allow for a more accurate measurement of accuracy. The teacher might be biased to confirm a feedback suggestion as it requires less effort than providing a new comment. We noticed that most teachers took the review of the automatic feedback suggestions seriously, but we cannot guarantee that some of the 49 involved teaching assistants failed to fully review the automatic feedback suggestions.

Two of the authors of this paper have been involved in teaching the course SE1 and might have influenced the empirical evaluation. However, we tried to clearly separate the research and instructor perspective. Two additional instructors have been involved in the course SE1 who are not authors of this paper, and the third author of the paper reviewed the results carefully without being involved in the course. In addition, we observed similar results in the second course, which was taught by an independent instructor who was not involved in the research.

External Validity: Most analyzed exercises have been in the domain of software engineering and computer science in the same university. While we believe that the approach is generalizable for other domains, we have not shown this in this study.

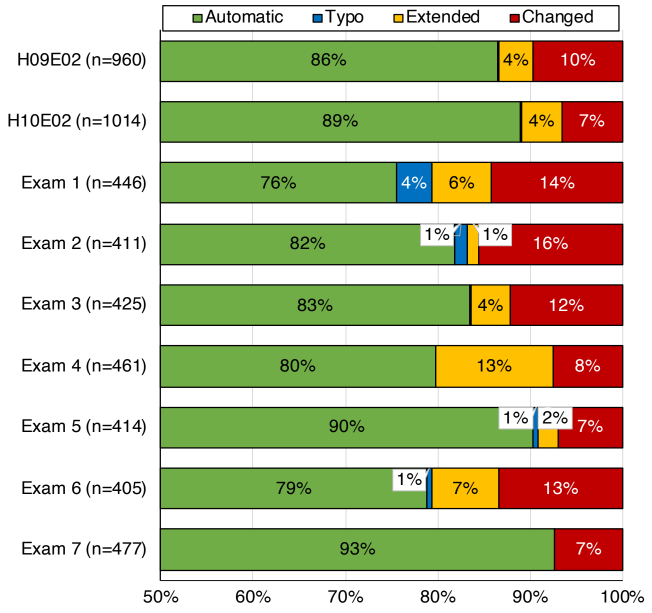


Figure 9. On average, 85% of computer-aided feedback comments remained unchanged (green) or only included minor typo fixes (blue). Furthermore, 5% were extended (yellow), and 10% were changed (red).

Construct Validity: The validity of the ratings might be affected by the wording of the question and by the score that the students received. Students with a higher score are typically more satisfied and less likely to complain about the quality of the feedback. Therefore, a good rating does not necessarily mean that the feedback had a good quality. Another limitation could be the fact that students like the approach of getting feedback. The ratings measure the perceived quality which is subjective. We can only infer the quality based on the ratings. Therefore, we consider Finding 3 on the quality of the ratings as anecdotal evidence.

Discussion

The review coverage of Athene is higher for exercises that do not ask for examples, but rather require students to work based on a given example. In the exercises `Exam 1` and `Exam 3`, students were asked to extract requirements or use cases from a given problem statement. In those exercises, the coverage was above the average with 70% and 50%, respectively. These questions still require students to apply problem-solving skills, but limit the variability of the answers. This leads to more similar answers and more reusable feedback.

Exercises asking for examples, such as the SE1 homework exercises, have lower review coverage of between 17% and 36%. This may be due to the increased variability of answers with different examples. As Athene tries to find similar text segments, it is more difficult to find a group with shared segments as students can describe all possible examples. Therefore, it is less likely to find reusable feedback among students.

Athene reuses reviews from teachers. The quality of the feedback suggestions depends on the quality of the manual feedback provided during the teacher reviews. If teachers provide incorrect manual feedback, Athene will not be able to pro-

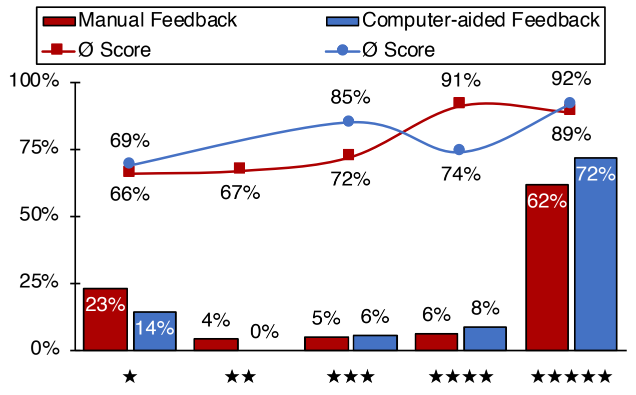


Figure 10. All ratings for SE1 homework (HXX) exercises by star rating. In this figure, ratings are grouped by the assessment type *Manual* ($n =$ 325) **or *Computer-aided*** ($n = 71$)**. The average score in percent is depicted per rating and assessment type. In the study,** 396 **out of** 10,240 **reviews were rated by students.**

vide correct feedback suggestions. In the example of SE1, the teachers who review the submission consist primarily of teaching assistants, who have limited experience in grading or providing feedback.

Nevertheless, the approach can improve the review process as it allows instructors to handle larger amounts of reviews or to inspect examples. Other systems presented in the Related Work section suggest comparing answers only with a sample solution provided by an instructor [25], thus reducing the variability in the solution space, which might limit the creativity of the students. However, creativity is an important aspect in software engineering education [14].

CONCLUSION

This paper presents three main contributions:

1. The machine-learning based **approach** CoFee was presented to suggest feedback for textual exercises. The approach is based on segmentation and similarity-based clustering. It reuses feedback on segments within the same cluster and learns which aspects of student answers are correct during the assessment.
2. Athene, a **reference implementation** of CoFee, using the ELMo language model and the HDBSCAN clustering algorithm was presented. Athene is integrated into Artemis and published as open-source software under the MIT license.[7]
3. An **empirical evaluation** of Athene in two courses with 2,300 students and 53 teachers in 17 textual exercises was conducted. The results of the quantitative evaluation in these exercises show that Athene can suggest up to 70% of the feedback with an average accuracy of 85%. Ratings provide first indications that the quality improves when compared to purely manual assessments.

The evaluation also shows that these numbers depend on the type of the textual exercise and on the variability of the possible solutions. A higher variance in correct solutions leads to less coverage because of fewer similarities in the student answers.

Athene does not require training data before the reviewing process to learn correct answers and feedback suggestions. Instead, it collects knowledge during the assessment. This incremental process allows instructors to change or introduce new exercises as needed, preventing students from submitting solutions from previous years. However, when reusing past exercises, Athene could profit from additional knowledge captured in these reviews. Future work needs to evaluate whether training data from the same exercise in previous years can improve the coverage or accuracy of feedback suggestions.

[7]Athene: https://github.com/ls1intum/Athene

REFERENCES

[1] Carlos Alario-Hoyos, Carlos Kloos, Iria Estévez-Ayres, Carmen Fernández-Panadero, Jorge Blasco, Sergio Pastrana, and J Villena-Román. 2016. Interactive activities: the key to learning programming with MOOCs. *European Stakeholder Summit on Experiences and Best Practices in and Around MOOCs* 319 (2016).

[2] Lorin W. Anderson, David R. Krathwohl, Peter W. Airasian, Kathleen A. Cruikshank, Richard E. Mayer, Paul R. Pintrich, James Raths, and Merlin C. Wittrock. 2001. *A Taxonomy for Learning, Teaching, and Assessing: A Revision of Bloom's Taxonomy of Educational Objectives*. Longmans Green.

[3] Sumit Basu, Chuck Jacobs, and Lucy Vanderwende. 2013. Powergrading: a clustering approach to amplify human effort for short answer grading. *Transactions of the Association for Computational Linguistics* 1 (2013), 391–402.

[4] Jan Philip Bernius. 2021. Toward Computer-Aided Assessment of Textual Exercises in Very Large Courses. In *52nd ACM Technical Symposium on Computer Science Education (SIGCSE '21)*. 1386.

[5] Jan Philip Bernius and Bernd Bruegge. 2019. Toward the Automatic Assessment of Text Exercises. In *2nd Workshop on Innovative Software Engineering Education*. Stuttgart, Germany, 19–22.

[6] Jan Philip Bernius, Anna Kovaleva, and Bernd Bruegge. 2020a. Segmenting Student Answers to Textual Exercises Based on Topic Modeling. In *17th Workshop Software Engineering im Unterricht der Hochschulen (SEUH)*. Stuttgart, Germany, 72–73.

[7] Jan Philip Bernius, Anna Kovaleva, Stephan Krusche, and Bernd Bruegge. 2020b. Towards the Automation of Grading Textual Student Submissions to Open-ended Questions. In *European Conference on Software Engineering Education*. ACM, 61–70.

[8] John Biggs. 2003. Aligning teaching and assessing to course objectives. *Teaching and learning in higher education: New trends and innovations* 2 (2003), 13–17.

[9] Benjamin S. Bloom, Max D. Engelhart, Edward J. Furst, Walker H. Hill, and David R. Krathwohl. 1956. *Taxonomy of educational objectives. The classification*

of educational goals. Handbook 1: Cognitive domain. Longmans Green.

[10] Richard P. Feynman. 1994. *Six Easy Pieces*.

[11] Richard Higgins, Peter Hartley, and Alan Skelton. 2002. The conscientious consumer: Reconsidering the role of assessment feedback in student learning. *Studies in higher education* 27, 1 (2002), 53–64.

[12] Alastair Irons. 2007. *Enhancing learning through formative assessment and feedback*. Routledge.

[13] Paul Kirschner, John Sweller, and Richard Clark. 2006. Why minimal guidance during instruction does not work: An analysis of the failure of constructivist, discovery, problem-based, experiential, and inquiry-based teaching. *Educational psychologist* 41, 2 (2006), 75–86.

[14] Stephan Krusche, Bernd Bruegge, Irina Camilleri, Kirill Krinkin, Andreas Seitz, and Cecil Wöbker. 2017a. Chaordic Learning: A Case Study. In *39th International Conference on Software Engineering: Software Engineering Education and Training*. IEEE, 87–96.

[15] Stephan Krusche and Andreas Seitz. 2018. ArTEMiS: An Automatic Assessment Management System for Interactive Learning. In *49th ACM Technical Symposium on Computer Science Education (SIGCSE)*. 284–289.

[16] Stephan Krusche and Andreas Seitz. 2019. Increasing the Interactivity in Software Engineering MOOCs - A Case Study. In *52nd Hawaii International Conference on System Sciences*. 1–10.

[17] Stephan Krusche, Andreas Seitz, Jürgen Börstler, and Bernd Bruegge. 2017b. Interactive learning: Increasing student participation through shorter exercise cycles. In *19th Australasian Computing Education Conference*. ACM, 17–26.

[18] Stephan Krusche, Nadine von Frankenberg, and Sami Afifi. 2017c. Experiences of a Software Engineering Course based on Interactive Learning. In *Tagungsband des 15. Workshops Software Engineering im Unterricht der Hochschulen (SEUH)*. CEUR, 32–40.

[19] Stephan Krusche, Nadine von Frankenberg, Lara Marie Reimer, and Bernd Bruegge. 2020. An interactive learning method to engage students in modeling. In *International Conference on Software Engineering: Software Engineering Education and Training*. 12–22.

[20] Vladimir I. Levenshtein. 1966. Binary Codes Capable of Correcting Deletions, Insertions, and Reversals. *Soviet Physics-Doklady* 10, 8 (1966), 707–710.

[21] Yang Li and Tao Yang. 2018. *Word Embedding for Understanding Natural Language: A Survey*. Springer International Publishing, Cham, 83–104.

[22] Leland McInnes and John Healy. 2017. Accelerated Hierarchical Density Based Clustering. In *International Conference on Data Mining Workshops*. 33–42.

[23] Tom Mitchell, Terry Russell, Peter Broomhead, and Nicola Aldridge. 2002. Towards robust computerised marking of free-text responses. (2002).

[24] Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global Vectors for Word Representation. In *Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Doha, Qatar, 1532–1543.

[25] Diana Perez, Alfio Gliozzo, Carlo Strapparava, Enrique Alfonseca, Pilar Rodríguez, and Bernardo Magnini. 2005. Automatic Assessment of Students' Free-Text Answers Underpinned by the Combination of a BLEU-Inspired Algorithm and Latent Semantic Analysis. 358–363.

[26] Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep Contextualized Word Representations. In *Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 2227–2237.

[27] Karl Raimund Popper. 1972. *Objective Knowledge*.

[28] Juan Ramos. 2003. Using TF-IDF to Determine Word Relevance in Document Queries. In *1st instructional conference on machine learning*, Vol. 242. Piscataway, NJ, 133–142.

[29] Per Runeson, Martin Höst, Austen Rainer, and Björn Regnell. 2012. *Case Study Research in Software Engineering*. John Wiley & Sons, Inc.

[30] Arjun Singh, Sergey Karayev, Kevin Gutowski, and Pieter Abbeel. 2017. Gradescope: A Fast, Flexible, and Fair System for Scalable Assessment of Handwritten Work. In *4th Conference on Learning @ Scale*. ACM, 81–88.

[31] Jana Sukkarieh, Stephen G Pulman, and Nicholas Raikes. 2003. Automarking: using computational linguistics to score short, free-text responses. (2003).

[32] Reed Williams and Thomas Haladyna. 1982. Logical Operations for Generating Intended Questions (LOGIQ): A typology for higher level test items. *A technology for test-item writing* (1982), 161–186.

[33] William E. Winkler. 1990. String Comparator Metrics and Enhanced Decision Rules in the Fellegi-Sunter Model of Record Linkage. In *Section on Survey Research*. 354–359.