Mining Massive Open Online Courses (MOOCs) for Software Testing Knowledge

Neda Džiugaitė[1]

Supervisor(s): Andy Zaidman[1]**, Baris Ardic**[1]

[1]EEMCS, Delft University of Technology, The Netherlands

A Thesis Submitted to EEMCS Faculty Delft University of Technology,
In Partial Fulfilment of the Requirements
For the Bachelor of Computer Science and Engineering
June 25, 2023

Name of the student: Neda Džiugaitė
Final project course: CSE3000 Research Project
Thesis committee: Andy Zaidman, Baris Ardic, Koen Langendoen

Abstract

Software testing is a necessary aspect of software development. With high expectations placed on software testers and a shortage of qualified professionals, Massive Open Online Courses (MOOCs) have emerged as a potential solution to improve software testing education. MOOCs provide accessible education and can offer a comprehensive review of software testing principles and procedures, bridging the gap between formal education and industry expectations. A study of software testing MOOCs was conducted to examine key aspects and compare concepts with university curricula and industry expectations. The findings show that a MOOC on average covers more concepts than a single university course. Additionally, MOOCs align well with what the industry expects from software testing practitioners. Therefore, MOOCs can successfully contribute to software testing education and bridge the gap between university curricula and industry expectations.

1 Introduction

Software testing is necessary to assure the dependability and quality of software programs. Software testing encompasses many different practices and techniques, various types of testing and many tools used to aid the process. Florea and Stray found that software testers are not expected to specialize in one particular activity, such as performance, automation, or test management, but rather to be proficient in a wide range of testing-related skills [1]. High expectations put on practitioners makes it not surprising that there is a lack of qualified professionals in the software testing field [2]. Zhu and Zhang identified that one of the problems in software testing education is that formal education does not align with the industry demand [2]. Massive Open Online Courses (MOOCs) can be one of the solutions to bridge the gap between formal education and industry expectations. MOOCs have been a popular medium to learn new skills or enhance one's existing skillset since their emergence in 2006[3]. MOOCs can give people who have a market need, access to high-quality education at a fraction of the cost [4]. They can provide accessible education in a variety of disciplines. MOOCs can offer an organized and thorough review of software testing principles and procedures and allow practitioners to fill any knowledge gaps they may have on technical subjects to meet industry expectations.

This paper aims to provide insight into how MOOCs contribute to software testing knowledge and education. To better understand the role of MOOCs in this field, we conducted a study of MOOCs about software testing. The results of the study were then used to compare concepts discussed in software testing MOOCs to university curricula and industry expectations.

RQ1 What are the key aspects of software testing MOOCs?

- **RQ1.1** What information is provided for the user before starting a MOOC?
- **RQ1.2** What are the entry requirements for software testing MOOCs?
- **RQ1.3** What teaching techniques are most common in software testing MOOCs?
- **RQ1.4** What are the most common software testing concepts discussed in MOOCs?

RQ2 Do the concepts taught in MOOCs align with what is being taught in universities?

RQ3 Do the concepts taught in MOOCs align with what the industry expects from software testing practitioners?

The remainder of the paper is laid out as follows: Section 2 provides related work. Section 3 presents the research method of the study. Section 4 focuses on the results of the study. A discussion about the findings is presented in Section 5, followed by threats to validity. Section 6 includes the conclusion and future work. Finally, Section 7 discusses aspects of responsible research of this study.

2 Related Work

We structure the related work around 2 main topics. These topics include software testing education together with industry expectations of software testing practitioners and MOOCs.

2.1 Software Testing Education and Industry Expectations

This study will look into the most common software testing concepts discussed in MOOCs. There has been research done about the most common software testing topics and skills in other mediums and places such as university courses and job advertisements for software testing positions.

Ardic and Zaidman carried out a study to better understand how higher education teaches aspiring software engineers about software testing [5]. They examined university curricula to see what are the most commonly discussed software testing concepts in dedicated software testing courses in universities. They found that "test process" and "test type" testing skill categories are the most common among software testing university courses.

Florea and Stray examined what skills, educational attainment, certified qualifications, and previous experience software testers are expected to have according to industrial demand [1]. They analyzed job advertisements to find the most common software testing skills, concepts, and tools that employers in the software testing industry required. Their study demonstrates that software testing is a position that requires a large number of specific skills.

Cerioli et al. mined 5 million job advertisements to find what types of testing and what tools and frameworks are most often required in the industry for coders and testers [6]. They found that software testing is essential to the sector. Furthermore, learning how to use automated testing tools like Selenium is advantageous because businesses prefer automated testing to manual testing.

Kassab et al. examined 1000 job postings in order to study the fundamental responsibilities that a software tester in the

US market is required to fulfil [7]. They looked at the industrial demand for practitioners in terms of testing skills, technical skills, soft skills, and educational attainment. They found that regression testing is the most commonly required testing type. In addition, they found out that automatic testing was more in demand than manual testing. Predictably, they found a high demand for testing automation tools as well.

2.2 MOOCs

There are many studies done on MOOCs, focusing on various aspects. The most relevant of which will be mentioned here.

Sharov et al. conducted quantitative research to examine online Python programming courses offered on well-known MOOC platforms [8]. Based on the large number of courses at various entry levels they concluded that MOOCs are a good medium to learn the Python programming language. Similarly to this study, they analyze MOOCs on a certain subject, however, they focus on quantitative measures such as the number of courses per platform, entry level, price, duration etc., whereas this study will go more in-depth into each course, beyond just quantitative indicators.

Bali examined a few MOOCs to see what pedagogical practices are common in MOOC education and suggest improvements [9]. They conclude that the advantages of MOOCs come more from the engagement that a course provides rather than from the way the MOOCs are constructed or from what the instructor assigns learners. Koedinger et al. conducted a study to examine what learning practices are common in MOOCs and how effective they are [10]. According to their findings, offering more interactive activities will improve student learning outcomes while video lectures may not add much to the learning process. Both these studies focused on the teaching techniques used in the selected MOOCs.

Ma et al. did a systematic literature review to figure out what factors influence the purchase of online courses [11]. They examine existing literature to find the relationship between course-related factors, instructor-related factors and platform-related factors and the purchases of MOOCs. From this study, we can observe what factors the user finds important.

3 Methodology

This paper aims to investigate key aspects of software testing MOOCs. To achieve this, a few MOOC providers were selected and data was collected from selected courses. This section outlines motivations for provider and course selection and explains how we collected data from each course.

3.1 MOOC provider selection

The providers were selected based on the list of free online course platforms provided by the European Job Mobility Portal (EURES) [12]. In addition to those six providers, we also considered Udacity, because it is another popular MOOC provider listed on various blogs [13; 14; 15]. The selected providers are Coursera, EdX, FutureLearn, Udemy, Saylor, Khan Academy and Udacity.

3.2 Course selection

MOOCs were selected based on the search functionality provided by the provider platforms. In-depth details of search terms, filters, and sorting options used can be found in the replication package [16]. In order to narrow down the search, additional filters and sorting options were used. Only MOOCs dedicated to software testing were considered for this study. This means that more general software engineering MOOCs that may include a section on testing were not taken into account. Additionally, we only considered courses in English. Since it was not feasible to analyze all of the software testing courses, we used heuristics that aided in choosing the most popular and liked MOOCs. These heuristics vary for different providers based on the available filters. An overview of the course selection for each provider can be found in Table 1.

EdX, FutureLearn, Udacity, Saylor and Khan Academy offer a smaller number of courses. After searching for "software testing" in the search bar and applying filters where possible, the platforms provided 20 courses or fewer. It was possible to manually go through all of the courses and select which are dedicated to software testing and could have been used for the study. This resulted in 3 courses from EdX, 3 courses from FutureLearn, 1 course from Udacity and 0 courses from Saylor and Khan Academy.

Coursera offers a larger number of courses. To narrow down our results we followed the same method of search functionality and filters. We manually went through 72 MOOCs to select courses that were explicitly dedicated to software testing and then 3 courses with the highest ratings were chosen. Courses which did not have any reviews or had their rating hidden were not considered. It is worth noting that Coursera offers projects and guided projects which take less than 2 hours to complete and offer hands-on activities to learn a specific tool or skill. These were not considered for this study, since they are very narrow in terms of content and only cover a single topic or tool.

Udemy offers the largest number of courses out of the selected providers. After searching for software testing courses, 10000 courses were shown. In order to select the most popular and liked courses, only the courses with a rating of 4.5/5 or higher were considered. This still yielded too many results. Therefore, the courses were sorted by the most reviewed and the first 3 dedicated software testing courses were chosen.

Most of the providers do not offer more than 3 dedicated software testing MOOCs. Therefore we decided to choose no more than 3 courses for each provider to analyze the course descriptions, entry requirements and teaching techniques. These factors could partially be influenced by the provider's platform and functionality. While analyzing the courses we noticed that each platform tends to present its courses in a similar way, therefore, course descriptions are dependent on the format that the platform uses. Entry level can be subjective and vary from provider to provider. Lastly, teaching techniques available in courses depend on what functionality the provider platform has, for example, it seems that discussion prompts are not available on some platforms. Because of these reasons having an equal distribution of courses among the provider gives less biased results.

Provider	Results after search and filters	Dedicated software testing MOOCs
Coursera	72	3+5*
EdX	20	3
FutureLearn	20	3
Udemy	3141	3+5*
Saylor	7	0
Khan Academy	7	0
Udacity	7	1

* - There were more dedicated software testing MOOCs, therefore heuristics were used to select the courses.

Table 1: Course selection from each of the 7 providers.

We decided to add additional courses when analyzing the topics since these are not influenced by the platform the course is on. Therefore, we added 10 more courses: 5 from Coursera and 5 from Udemy. Additional courses from Coursera were selected in the following way. We used the same filters as before, however, we selected the first 5 dedicated software testing courses, disregarding the reviews. This way courses that do not offer ratings or reviews can also be taken into account. Additional courses from Udemy were selected in the same way as explained before.

3.3 Data collection

In order to find the most commonly discussed concepts in software testing MOOCs we analyzed each of the courses that were selected for this study. We noted some metadata such as price, duration, level, and prerequisites. When the entry level was missing, we derived it from prerequisites or course descriptions. Additionally, we took note of ratings, the number of ratings and the number of people enrolled in the courses whenever this data was available.

Furthermore, we analyzed what data is available for the user before starting a course. All of the information for this part was gathered without accessing the course itself, only from what is presented before enrolling.

Moreover, the teaching techniques used in the courses were analyzed. We noted whether courses include videos, reading material, quizzes, practical exercises and discussion prompts. There were 2 courses that were behind a paywall and 2 courses that were supposed to start in the future and were not accessible at the time of conducting this research. Therefore, it was not possible to collect data about the teaching techniques for 4 out of 13 courses.

Lastly, we collected data about what topics and tools were taught in each course. We selected a total of 23 courses for topic analysis. Keywords regarding topics and tools were gathered from the syllabus, course content, learning objectives and course descriptions or a subset of these if some aspects were missing or inaccessible. The majority of the time keywords were extracted from video or reading section titles and descriptions. When the titles were too vague or descriptions were missing we extracted topics from the first or last video of the module, when the instructor outlined the topics or did a recap of the module.

4 Results of the MOOC analysis

This section will discuss the results of the study that was conducted. The full dataset with all of the results can be found at [16].

4.1 Course descriptions

During our research, we observed what information was presented to the user before starting a course. It was noted whether the course description provides reviews, learning objectives, prerequisites, syllabus and teaching techniques.

The results can be found in Figure 1. It can be observed that almost all courses provide learning objectives, prerequisites and syllabi, however, these vary in detail. Most courses have fairly vague prerequisites, some of which are inaccurate as mentioned in course reviews or observed after analyzing the course content. Seven of the courses mention what teaching techniques are used in the course. Although, the teaching techniques in the descriptions are often not complete. There are no mentions of discussion prompts or peer-graded assignments. Additionally, it is rarely mentioned what labs are available if any. Lastly, reviews are only available for 5 of the courses. This is due to the fact that only Coursera and Udemy have review functionality for the courses in this study. Even among these 2 providers, there was 1 course which did not allow the user to see the reviews, the user could only observe the course rating.

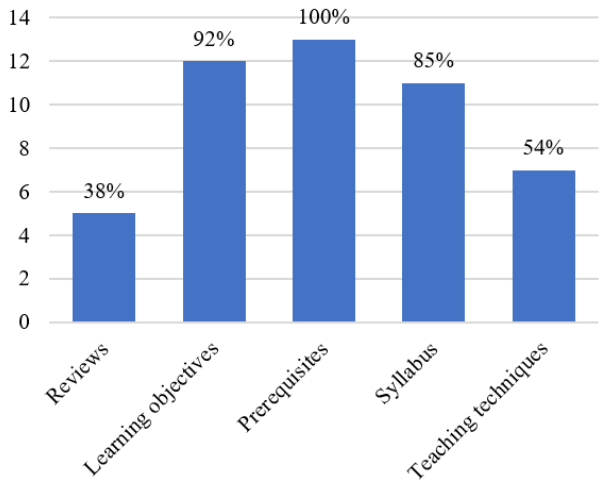

Figure 1: Results of what is presented for users in the course descriptions of software testing MOOCs before enrolling.

4.2 Entry requirements

While conducting the study we noted the entry level of the courses as well as the prerequisites. Eight of the courses state the entry level explicitly. The levels include beginner, intermediate and advanced. The level for the other 5 courses was concluded from the prerequisites listed.

Four of the 13 courses were tagged as beginner courses, 8 courses were of intermediate level and 1 course was advanced. Beginner courses have little to no prerequisites,

most of them list basic computer skills as the main prerequisite. The prerequisites for intermediate courses vary a bit more. Half of these courses require or recommend the student to have knowledge of a certain programming language. Other prerequisites include the ability to install and run an IDE, mathematical and logical reasoning and shell scripting. Lastly, the advanced course is a follow-up on one of the intermediate courses, therefore the prerequisites are to be familiar with the outline of the first course and the student should already have practical experience with testing.

4.3 Teaching techniques

Additionally, data was gathered regarding the teaching techniques used in software testing MOOCs. We checked whether courses provide videos, readings, quizzes, practical exercises and discussion prompts. Any other teaching techniques found were also noted.

As can be seen in Figure 2 videos are the most popular teaching technique with 8 out of 9 courses using them. Reading and practical exercises are commonly employed as well. Quizzes are the less popular teaching technique with only 5 courses providing them. Additionally, a few courses make use of discussion prompts and peer-graded assignments. Discussion prompts allow students to post their thoughts on a prompt on a message board in the platform and discuss with peers or instructors. Practical exercises are provided in various ways. Half of the courses that provide practical exercises provided them in a dedicated lab environment. Only one course offered automatic grading, while, others usually provided example solutions.

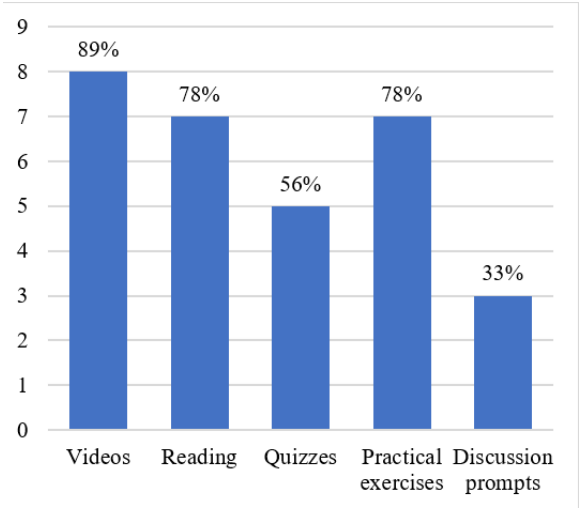

Figure 2: Results of the teaching techniques found in software testing MOOCs.

4.4 Topics covered

Lastly, we extracted keywords from the learning objectives, syllabus, course descriptions and course content to gather data about software testing concepts discussed in the courses. It can be concluded that the most common keywords in software testing MOOCs are automation (61%), unit testing (57%), test levels (52%) and test coverage (43%).

In order to have a better overview of the data collected, it was decided to categorize the keywords. For that, we will be using the taxonomy of software testing skills as defined by Florea and Stray [1]. In addition to the skills listed in the taxonomy, it is necessary to clarify how we classified keywords that were not listed in the taxonomy. Keywords such as automation, test-driven development, and behaviour-driven development were included in the testing process category. Various types of testing such as mutation testing, performance testing, load testing, API testing, and so on were included in the test type category.

The results after categorization can be found in Figure 3. Almost all courses covered at least some aspect of the test process category. More than half of the MOOCs covered test levels, test types and various tools used in software testing. The least covered topic was static testing.

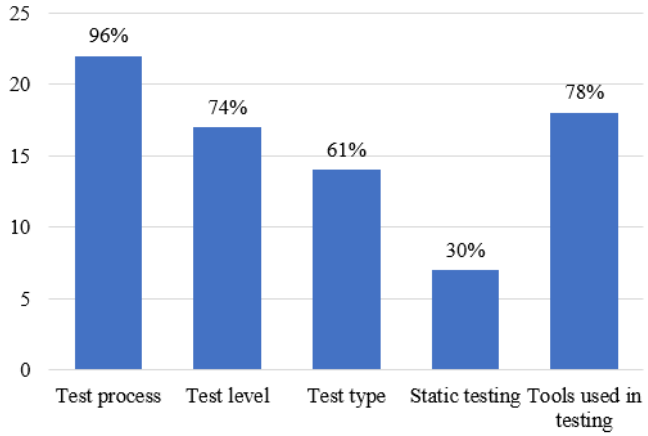

Figure 3: Results of the concepts discussed in software testing MOOCs, categorized using the taxonomy of software testing skills[1].

4.5 Comparing results

We compared our findings of the most commonly discussed software testing aspects in MOOCs to the ones in dedicated software testing university courses and to requirements from software testing job adverts. We used research done by Ardic and Zaidman [5]. They analyzed university curricula in order to figure out what software testing topics are commonly discussed in dedicated software testing courses. Their results are presented in a similar way, as they use the same software skills taxonomy by Florea and Stray [1]. Furthermore, we compared results with a study by Florea and Stray, who defined the software skills taxonomy used in this paper[1]. They analyzed job advertisements in order to find what software testing skills are most in demand in the industry. The comparison can be found in Figure 4.

The categories of test process, tools used in testing and test level are most covered by MOOCs. The difference in coverage for test process and tools is higher for MOOCs and universities than it is for MOOC and industry needs. Test levels

are much more popular in MOOCs than they are in universities or required in job advertisements.

The test type category has similar coverage everywhere. MOOCs, university courses and job advertisements all cover test types in over 60% of the courses/adverts.

Static testing was the only category that has a significantly lower coverage in MOOCs compared to university courses. However, we see that static testing is rarely required in the industry.

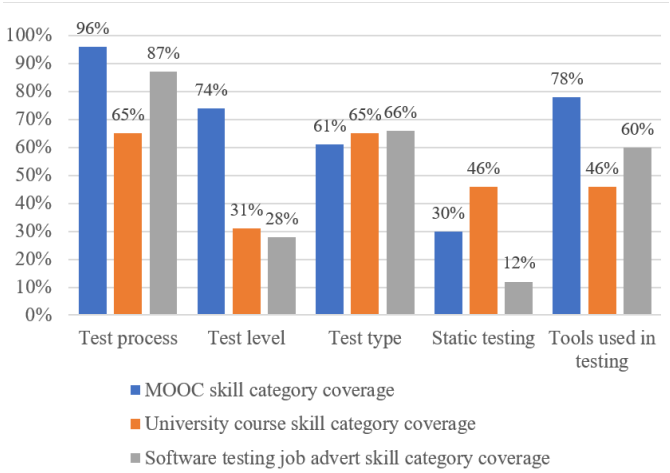

Figure 4: Results of the concepts discussed in software testing MOOCs, compared with what is taught in universities [5] and to software testing job advertisements [1], categorized using the taxonomy of software testing skills [1].

5 Discussion

This section revisits the research questions and discusses the findings, comparing them to existing literature. Here, we also examine the possible threats to the validity of the study.

5.1 Revisiting the Research Questions

RQ1.1: *What information is provided for the user before starting a MOOC?* Information presented before enrolling is very important for paid courses, as naturally, the users would like to know what to expect before purchasing the course. However, around 85% of the courses analyzed in this study are free or have a limited free trial. Therefore users are still able to find the information they need before the purchase.

The majority of the courses provide the user with learning objectives, prerequisites and syllabi. Users tend to select online courses based on their expectations for learning outcomes as well as their needs for improving their present knowledge[17]. Therefore it is important to inform the user of the topics covered and level of training [8].

Information on what teaching techniques will be used was provided only in half of the courses. Teaching techniques in MOOCs are fairly predictable with the majority of the courses providing a combination of video content, reading material and practical exercises. However, there are exceptions. Therefore, it would be beneficial for users to know what to expect in terms of how the material will be presented. Additionally, users that are looking to gain more practical knowledge from these courses might find it beneficial to know what sort of practical exercises will be provided to them. This information was rarely provided without accessing the course.

Course reviews were fairly scarce with only 38% of the courses providing them. This might be due to the fact that only 2 out of the 7 selected providers have reviewing and rating functionality available for the courses in this study. Course reviews have a positive effect on enrollment and purchases of the MOOCs [11]. Hence, it was surprising that only a few courses provide reviews.

RQ1.2: *What are the entry requirements for software testing MOOCs?* The majority of the courses were of intermediate level and required the user to have knowledge of a certain programming language. Research shows that the majority of MOOC users already have a higher education degree [18]. Intermediate courses are appropriate to be able to bridge the gap between formal education and industry's need since students who would take these courses after their formal education would likely have experience with programming and should already be familiar with the basics of testing.

A few beginner courses, that require almost no prior knowledge, focus on high-level software testing knowledge. They introduce some software testing concepts but focus on teaching the students concepts like test planning, defect reporting, and test cases. These courses do not require prior software testing knowledge and can be taken without prior software engineering education.

The advanced course was a follow-up of one of the intermediate courses on the same platform. In order to take that course a student should be familiar with concepts discussed in the intermediate course and additionally already have some practical knowledge of software testing. This course would likely be suitable for someone who has already taken a course on software testing course in a university or elsewhere.

RQ1.3: *What teaching techniques are most common in software testing MOOCs?* Most of the courses provide video lectures, reading material and some form of practical exercises. Watching videos and reading contributes to the learning outcomes in a limited way, whereas practical exercises contribute more significantly to the learning results [10]. Around half of the courses provide quizzes to assess the student's knowledge throughout the course. Quizzes are a form of overly simple assessment that does not accurately reflect the real world [9]. Therefore, it is great to see that most courses employ more complex practical exercises, instead of, or alongside quizzes.

Since MOOCs are aimed at large audiences, interaction with instructors or faculty is oftentimes not feasible. Discussion prompts and forums as well as peer-graded exercises provided in a few of the courses allow students to interact with others and cooperate. Students can respond to the discussions faster than most instructors, therefore, discussion prompts can help students reach a correct answer without the instructor's intervention [9]. Therefore, discussion forums on peer-graded assignments are a good way to address the lack of interaction between the instructor and students, and more

courses should employ these teaching techniques.

RQ1.4: *What are the most common software testing concepts discussed in MOOCs?* After the keyword extraction, the most commonly discussed topics were found to be automation, unit testing, test levels and test coverage. Software testing can be expensive with 30% to 60% of all life-cycle costs spent on testing [19]. The need to automate some of the expensive processes including testing is understandably appealing, therefore, it makes sense that automation is a popular teaching topic. Other topics mentioned above cover some testing basics, therefore necessary for most software testing courses.

After categorizing the keywords, we find that almost all courses cover some aspect of the testing process and more than half of the courses cover test levels, various test types and tools or frameworks used for testing. All of these categories together are necessary to be able to properly conduct software testing. The least covered topic is static testing. Static testing has some limitations when it comes to recall, precision or performance, which might contribute to this type of testing being used less [20]. Hence, that might be a reason why it is covered less in software testing MOOCs.

RQ2: *Do the concepts taught in MOOCs align with what is being taught in universities?* The topic coverage for most categories in MOOCs is higher than in universities, meaning that percentages are similar or higher for MOOCs. This indicates that on average, a single software testing MOOC covers more different topics than a dedicated software testing course. However, it is unclear how in-depth these topics are covered. Software testers are required to be proficient in a wide array of different software testing skills, hence higher topic coverage is beneficial for someone preparing to go into industry[1].

RQ3: *Do the concepts taught in MOOCs aligns with what the industry expects from software testing practitioners?* In terms of software testing skill categories, MOOCs align with what the industry needs from software testing practitioners. Most of the software skills asked for in job advertisements are covered in MOOCs with similar or higher percentages.

It was found that automation is the most common keyword among software testing MOOCs. Automation is especially in demand in the industry [7; 6; 1]. The most often required testing type in the industry is regression testing [6; 7]. We found that regression testing is covered by 26% of MOOCs. This means that regressions testing was the most popular software testing type in the analyzed MOOCs. Other popular types of testing were performance (22%), functional (17%) and security (17%). These types were also popular among job advertisements [6; 7]. Selenium is by far the most popular software testing tool covered in MOOCs with a 30% coverage. This tool is also the most often required of software testing practitioners [6; 7; 1].

To conclude, the content of MOOCs aligns with the industry demands in terms of software testing skill categories, keywords, testing types and tools. A part of MOOCs' target audience is practitioners who want to increase their qualification [8]. Therefore, alignment between MOOC content and the industry's needs benefits both MOOC instructors and practitioners.

5.2 Threats to Validity

We recognize that there are potential threats to the validity of this study. This section will discuss threats to external validity, internal validity and construct validity.

External validity. This study provides an analysis of software testing MOOCs. We were able to analyze 13 courses for course descriptions, entry requirements and teaching techniques and 23 courses for software testing topics. All the courses were selected from 7 MOOC platforms. This relatively small selection of courses might not accurately reflect the wide range of software testing MOOCs available, which could skew results. Therefore, even if certain general tendencies are shown among the most popular MOOCs, we cannot generalize the findings of this study to all software testing MOOCs.

Internal validity. When selecting the courses we employed a systematic strategy with specified search terms, filters and sorting options that were used to ensure that the study could be repeated, as shown in the replication package [16]. However, it is possible that we missed some relevant courses because of the filters chosen. We exhausted 5 of the 7 MOOC providers for dedicated software testing courses and focused on the popularity of the courses for the other 2 providers. Since we focused on the popularity of the providers and the courses, the search results we achieved would be similar to the search results of the average user looking for a software testing MOOC. Therefore, the missing relevant MOOC rate should be low.

Additionally, it is possible that a change in course ratings or content could have occurred. Consequently, this could alter which courses should have been included in or excluded from the sample. To mitigate this threat, we took a snapshot of the MOOCs in our sample and the replication package includes the data collection dates.

Construct validity. It is important to recognize the potential drawbacks associated with relying simply on the details provided in the MOOC descriptions and content overview to analyze the courses. These descriptions could lack information regarding the topics taught, teaching techniques and entry requirements, therefore, it is possible to miss keywords, concepts or techniques. Additionally, we recognize that the self-described level of difficulty, such as "beginner" or "advanced," may not always correspond with the course's actual content. We clearly defined the methods of our data collection in Section 3.3, however, this threat is not fully mitigated in this study and could be addressed in future work.

6 Conclusion and Future Work

This work aimed to examine how Massive Open Online Courses (MOOCs) contribute to software testing knowledge and education. The study aimed to answer research questions related to the key aspects of software testing MOOCs, the alignment of concepts taught in MOOCs with university curricula, and the alignment of concepts taught in MOOCs with industry expectations of software testing practitioners' skills.

The results of our analysis show that dedicated software testing MOOCs are well-structured, providing users with necessary information such as learning objectives, prerequisites

and syllabus. The majority of the courses are of intermediate level which aligns with the fact that the majority of MOOCs' audience have higher education, therefore, they might already have some knowledge on the topic[18]. Teaching techniques used in software testing online education are more effective when there are practical exercises provided [10], which was the case for the majority of courses in this study. The most commonly discussed topic is automation, which is often required by the industry[1; 6; 7]. Lastly, the most common skill category is test process, which is also one of the most popular categories in university courses [5] and most often required in job adverts [1].

In terms of alignment between MOOCs and university curricula, we found that MOOCs cover most categories more often than they are covered by university courses. Similarly, higher percentages were found for MOOCs when compared to industry demands. This allows us to believe that a single software testing MOOC tends to cover a larger variety of topics than is covered by a university course or is expected of a single practitioner. Additionally, MOOCs align with industry expectations in terms of content. We found that the most popular keyword, tool and testing type in MOOCs are consistent with the most required by the industry.

This paper has analyzed MOOC's contribution to software testing education. The utilization of MOOCs is a possible solution to address the challenges of software testing education. By investigating the key aspects of software testing MOOCs and comparing them to university curricula and industry expectations, this study contributes to the advancement of software testing education and the professional development of practitioners.

Future work should aim to expand the sample size and consider additional selection criteria in order to have a more representative sample and a more comprehensive analysis. Furthermore, future research could consider employing additional methods for evaluating the courses, such as participant surveys or expert reviews, to obtain more reliable findings. Including multiple data sources would increase the robustness and validity of the results and ensure a more thorough analysis of software testing MOOCs.

7 Responsible Research

Reproducibility stands out as a key component of ethical research. Reproducibility is the capacity to produce the same outcome using the exact same methods, data, and tools. It also involves being able to confirm the outcomes using the same techniques and coming to the same conclusions using the same data [21].

While conducting the research presented in this paper, reproducibility was kept in mind at all times. This paper includes the summary of the results of the study conducted, however, there is also a full dataset available that includes all the data points. Additionally, this dataset includes general information about the MOOCs such as the provider and link. This way the reader is free to examine the data in the source and reproduce the results. In addition, the methodology section includes a description of how we analyse our results. This would allow someone to know how to reach the same outcome and conclusions using the same data.

One of the concerns is that some of the MOOCs analyzed in the study are not accessible. There are 2 courses that do not have a free trial. In addition, there were 2 courses whose content was not accessible at the time of writing this paper. To make sure that our research is reproducible, in these cases, we gathered data about these courses from the descriptions provided without enrolling for the course.

Another concern is that most of the courses are accessible for free only through a trial. This means that it might not be possible for some users to access the courses if they have already used a free trial on these MOOC platforms.

References

[1] R. Florea and V. Stray, "The skills that employers look for in software testers," *Software Quality Journal*, vol. 27, p. 1449–1479, 2019.

[2] B. Zhu and S. Zhang, "Curriculum reform and practice of software testing," in *Proceedings of the 2013 the International Conference on Education Technology and Information System (ICETIS 2013)*, pp. 846–849, Atlantis Press, 2013.

[3] W. Rahman, H. Zulzalil, I. Ishak, and A. Selamat, "Quality model for massive open online course (mooc) web content," *International Journal on Advanced Science, Engineering and Information Technology*, vol. 10, p. 24, 2020.

[4] I. Borrella, S. Caballero-Caballero, and E. Ponce-Cueto, "Taking action to reduce dropout in moocs: Tested interventions," *Computers & Education*, vol. 179, 2022.

[5] B. Ardic and A. Zaidman, "Hey teachers, teach those kids some software testing," in *Proceedings of the Fifth ICSE Workshop on Software Engineering Education for the Next Generation (ICSE SEENG)*, (Melbourne, Australia), pp. 9–16, IEEE, 2023.

[6] M. Cerioli, M. Leotta, and F. Ricca, "What 5 million job advertisements tell us about testing: A preliminary empirical investigation," in *Proceedings of the 35th Annual ACM Symposium on Applied Computing*, SAC '20, (New York, NY, USA), p. 1586–1594, Association for Computing Machinery, 2020.

[7] M. Kassab, P. Laplante, J. Defranco, V. V. G. Neto, and G. Destefanis, "Exploring the profiles of software testing jobs in the united states," *IEEE Access*, vol. 9, pp. 68905–68916, 2021.

[8] S. Sharov, S. Tereshchuk, A. Tereshchuk, V. Kolmakova, and N. Yankova, "Using mooc to learn the python programming language," *International Journal of Emerging Technologies in Learning (iJET)*, vol. 18, pp. 17–32, 2023.

[9] M. Bali, "Mooc pedagogy: Gleaning good practice from existing moocs," *Journal of Online Learning and Teaching*, vol. 10, no. 1, pp. 44–56, 2014.

[10] K. R. Koedinger, J. Kim, J. Z. Jia, E. A. McLaughlin, and N. L. Bier, "Learning is not a spectator sport: Doing

is better than watching for learning from a mooc," in *Proceedings of the Second (2015) ACM Conference on Learning @ Scale*, (New York, NY, USA), p. 111–120, Association for Computing Machinery, 2015.

[11] L. Ma, S. Pahlevan Sharif, and K. W. Khong, "What factors drive the purchase of paid online courses? a systematic literature review," *Journal of Marketing for Higher Education*, pp. 1–24, 08 2022.

[12] E. L. Authority, S. A. Directorate-General for Employment, and Inclusion, "Top 6 free platforms for online courses." Available: https://eures.ec.europa.eu/top-6-free-platforms-online-courses-2020-05-11_en, 2020. [Online].

[13] C. Ngo, "10 best free and affordable platforms for online courses," *BestColleges*, 2023. [Online]. Available at: https://www.bestcolleges.com/blog/platforms-for-online-courses/.

[14] EduTechReviews, "The 9 best moocs platforms," *Medium*, 2017. [Online]. Available at: https://medium.com/@edutechreviews/the-9-best-moocs-platforms-631846001b6e.

[15] G. Hekim, "The best moocs and e-learning platforms," *Dan Institute*, 2022. [Online]. Available at: https://daninstitute.com/blog/the-best-moocs-and-e-learning-platforms/.

[16] N. Džiugaitė, "Dataset of software testing massive open online courses (moocs)," 2023. [Online]. Available at: https://doi.org/10.4121/1c8546e4-ecfd-4c60-986e-42198d5e0d5b.v1.

[17] K. F. Hew and W. S. Cheung, "Students' and instructors' use of massive open online courses (moocs): Motivations and challenges," *Educational Research Review*, vol. 12, pp. 45–58, 2014.

[18] M. Rohs and M. Ganz, "Moocs and the claim of education for all: A disillusion by empirical data," *The International Review of Research in Open and Distributed Learning*, vol. 16, pp. 1–19, 2015.

[19] C. Ebert, *Global Software and IT: A Guide to Distributed Development, Projects, and Outsourcing*. Wiley-IEEE Computer Society Pr, 1st ed., 2011.

[20] P. Anderson, "The use and limitations of static-analysis tools to improve software quality," *CrossTalk-Journal of Defense Software Engineering*, vol. 21, 06 2008.

[21] A. Widener, "Reproducibility needs clearer terminology, paper says," *Chemical Engineering News*, 2016.