Topic Analysis on Popular Software Testing Books

Mining Software Testing Knowledge

Hasan Doruk Ozmetin[1]

Supervisor(s): Andy Zaidman[1]**, Barış Ardıç**[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: Hasan Doruk Ozmetin
Final project course: CSE3000 Research Project
Thesis committee: Andy Zaidman, Barış Ardıç, Koen Langendoen

Abstract

In this study, we try to understand what kind of topics and frameworks are covered by the popular software testing books, and see whether these topics satisfy the industry needs and address the rising trends. To define "popular" software testing books, we formulated three heuristics. The topics of the books are analyzed through LDA topic modelling and manual inspection. LDA results inform us on the dominance of the topics within the whole corpus, while the manual inspection results show how often a topic is addressed. We combine the results of both of the methods to analyse the most noteworthy topics. We found that test automation, test design and planning, coverage analysis were the most frequently and extensively discussed topics in our corpus. We conclude that although the books cover some major topics that are demanded by the industry, there are also areas such as test management and usability testing, which are underrepresented. We also observed that the popular software testing books do not cover the rising software testing trends. While JUnit was the most discussed framework, in general the software testing books do not include practical information for specific frameworks or tools, but rather focus on the tool selection process.

1 Introduction

As software becomes increasingly integrated into the daily lives of many, software testing gains a vital importance and place in maintaining the reliability of the software development products. To meet the need of designing credible and safe systems, the software engineers need to have strong skills in effective software testing [1]. What technical knowledge software engineers need to have about software testing and which information they actually acquire are critical concepts to study. Books, MOOCs and online resources are some of the ways the software engineers and testers can utilize to educate themselves. In this study, we are interested in particular the technical content that popular software testing books cover and whether their coverage of topics meet the industry needs and trends.

We aim to answer the main research question: "What are the major topics being discussed in popular software testing books?" through four sub-questions:

RQ1: What are some software testing topics that are commonly discussed in popular software testing books? Are there any patterns in terms of coverage of topics?

RQ2: How do the topics covered in popular software testing books compare to the topics that are most relevant to the industry needs and rising software testing trends?

RQ3: Are there any major areas of software testing that are underrepresented in popular books?

RQ4: What are the most common testing tools and frameworks discussed in popular software testing books?

We follow three main procedures to conduct this study:

- Collection of popular software testing books.
- Application of Latent Dirichlet Allocation (LDA) topic modeling technique to analyze the common patterns of topics throughout the corpus of books.
- Manual inspection of books and extraction of discussion topics and testing tools.

We first present the related work in Section 2, followed by the explanations of the data collection process and selected topic analysis methods in Section 3. In section 4, we present the findings from our results. In section 5, we discuss the responsible research principles. We discuss the results and the limitations of our methodology in Section 6. Finally in Section 7, we revisit our research questions, explain our conclusions and the implications of our study.

2 Related Work

Software testing skills that software engineers need to acquire is a well studied area. There are recent research that analyze the emerging trends in software testing, and the industry demands from software testers.

2.1 Software Testing Trends

Gurcan et al. apply LDA topic extraction method on an empirical corpus of research papers to evaluate the software testing trends in the past 40 years. They found out that predictive testing, security vulnerability and mobile application testing topics are among the most rising trends in the interval of years 2015-2020 [2].

Salahirad et al. used co-word analysis on software testing research to extract the pre-dominant topics and their relations. Based on the results they categorized the software testing field in 16 main topics and 18 sub topics. While "test oracles", "automated test generation", "evolution and maintenance" and "creation guidance" are topics that have the highest connections to the other topics in their topic node cluster model, the use of machine learning in software testing, web and application testing, automated test generation and program repair are the emerging trends in the software testing research [3].

Boukhlif et al. used bibliometric analysis on research based on intelligence testing in the interval from 2012 to 2022. Intelligence testing is the use of artificial intelligence (AI) and machine learning (ML) to improve the effectiveness and efficiency of the testing progress. It covers use of techniques such as automated test case generation with AI, predictive testing with ML, pattern recognition of test results to identify defect areas etc. Their findings show that the research on intelligence testing has a growth rate of 14.87% every year [4].

2.2 Industry Related

Florea and Stray have recently worked on research analyzing the software tester job advertisements. Their findings show that test design and planning, test automation, and performance testing are highly demanded areas and skills by the

employees [5][6].

More recently, Florea et al. used thematic analysis on interviews with software testers and software test job advertisements to understand the software tester roles in the industry. They identified 5 main software tester roles consisting of domain-specific tester, automation specialist, test infrastructure specialist, user experience tester, and test manager. The majority of the job advertisements demanded automation specialist and domain-specific testers [7].

Kassab et al. mined 1000 job advertisements related to software testing in the United States. They found that the test strategy planning, execution monitoring, test automation, acceptance testing are the most sought after skills for software testers. They also report that Selenium was the most demanded tool [8].

3 Methodology

This section discusses the methods followed for the book collection and topic analysis parts.

3.1 Data Collection

To define what a "popular" software testing book is, we formulated three heuristics, which we believe represent the popularity from a large range of aspects in the software testing field:

- *Frequency of use in top university computer science courses*: We picked this heuristic to capture the popular software testing resource books used in the academia. Many top universities in computer science field teach software testing, and academia is a significant part of software engineering education [23]. In the application of this heuristic we used the data gathered about the textbook usage in top universities by Ardic and Zaidman in their research [24].
- *Appearance in International Software Testing Qualifications Board (ISTQB) syllabi*: This heuristic helps us to include the books that are possibly preferred by the testers, who aim to start or advance their careers in the software testing industry. ISTQB is the "leading global certification scheme in the field of software testing" [25]. They provide internationally accepted certification exams in software testing. In this research their "foundations level", "foundation level agile tester", "advanced level test manager", "advanced level test analyst" syllabi were taken into account.
- *Appearance in Amazon best seller (top 100) in "Software Testing" category*: With this heuristic we aim to capture the preferences of a broader range of audience from hobbyists to more professional testers in the software testing field. This heuristic is the most dynamic one since the best sellers are subject to change each day. We used the best seller information at the time we conducted the collection process [26].

In the combination of these heuristics, we first treated each of them as a binary condition and added the books that satisfied more than one heuristic into our dataset. After that, in order to enlarge our dataset, we considered the top ranked books for each heuristic (excluding the already added ones). The rank condition was determined specific to each heuristic: for the Amazon best seller heuristics, a book was ranked higher if it was ranked highly by many users; for the ISTQB syllabi heuristic, the number of times the book occurred in different syllabi determined its rank, and for the university curriculum the number of times a book appeared in the curricula of the top universities was taken into account. We observed that after the addition of the books that satisfied more than one heuristic, majority of the books satisfied both the ISTQB and university curricula heuristics while there was a little representation of the Amazon Best Sellers heuristic. We wanted to capture the insights that each heuristic can bring into our dataset. Thus, to have a more balanced distribution for each heuristic in our dataset, we added the top 3 books that were ranked the highest in the Amazon Best Sellers heuristics (The Pragmatic Programmer, Clean Code, Site Reliability Engineering), while only the top books from both the ISTQB and university curricula heuristics. See Section 6 for further discussion on the addition of books. In the end we collected 14 popular books (which we found to be a good number to conduct our study within the given time frame) that represent a variety of software tester audience. The meta-data about the books including the author name, edition count, publication year are collected. The complete list of selected books and summary of metadata information can be found in Table 1. More information of our corpus is available in our replication package [27].

3.2 Data Analysis

We used Latent Dirichlet Topic Modelling (LDA), an unsupervised probabilistic topic modelling technique to analyze the common patterns of topics discussed overall in the collection of books. In LDA, each topic is represented by a distribution over words, and the documents of a corpus are represented as "random mixture of latent topics" [28]. Additionally, each book has been manually inspected to catch semantics that LDA did not discover, gain more detailed insights on the topic coverages and extract the tools and frameworks that are mentioned. We use LDA and our manual inspection results in a complementary manner: inspection results show how often a topic was discussed, while LDA shows the dominance of the captured topics in the overall corpus. This way we are able to assess the topic coverage results more objectively.

LDA

In the application of LDA, a series of steps were taken consisting of corpus selection, corpus pre-processing and LDA implementation.

Corpus Selection: We chose the corpus to be composed of the chapters of the books rather than the whole books themselves. Representing a document in the corpus to be a chapter of a book enables to select only the related parts of the books, reducing the unnecessary noise in the corpus. Our corpus includes 192 documents (chapters). Since our

Book Name	Author(s)	Edition	Publication Date	Selected Chapters
Foundations of Software Testing [9]	Graham et al.	4th	2019	All except 7
Lessons Learned in Software Testing[10]	Kaner et al.	1st	2001	All except 9, 10
The Art of Software Testing [11]	Mayers et al.	3rd	2011	All except 1
Agile Testing [12]	Crispin & Gregory.	1st	2008	All except 3, 4
Introduction to Software Testing [13]	Ammann & Offutt	2nd	2016	All
Software Testing: A Craftman's Approach [14]	Jorgensen	4th	2013	All except 2
Systematic Software Testing [15]	Craig & Jaskiel	1st	2002	Except 8, 9, 10, 11, 12
The Pragmatic Programmer [16]	Thomas & Hunt	2nd	2020	13, 41, 42, 43
Clean Code [17]	Martin	1st	2008	9, 13, 15
Site Reliability Engineering [18]	Petoff et al.	1st	2016	see dataset
A Practitioner's Guide to Software Testing [19]	Copleand	1st	2004	All except 2
Software Testing and Analysis [20]	Pezze & Young	1st	2007	All
Software Testing [21]	Patton	2nd	2005	All except 22
Software Test Automation [22]	Graham	1st	1999	All

Table 1: Collected Books and Metadata

focus is on analyzing the technical aspects of software testing discussions in books, we only added a chapter if it directly discussed such topic (chapters that are solely based on the soft skills, case studies without insights on technical software testing topics or content unrelated to software testing were omitted). We detected such unrelated chapters by first identifying the possible chapters to be filtered out based on the chapter titles (e.g., "Cultural Challenges", "Your Career in Software Testing") for each book. After that we inspected those chapters and sub sections to see if the chapters are solely based on content that are out of the interest area of our study. The generic parts such as preface and bibliography sections are also filtered out. Selected chapters for each book can be found in the replication package and Table 1.

Corpus Pre-Processing: In order to apply LDA effectively, a pre-processing step is necessary. We first tokenized the words and applied lemmatization to convert them back to their root forms. After that, each token is lowercased; any punctuation, numbers and stop words are removed. The stopword list includes the conventional English stopwords and highly common keywords that occurred in each document such as "test", "testing", "software" etc. In order to identify the words that dominated the whole corpus, we applied an iterative process of ngram analysis. Top uni, bi and trigrams were compared and words, which dominated the ngram lists or did not form meaningful combinations in the context of software testing were removed. Lastly, the words of length smaller than 3 are filtered out as they do not add any significant meaning to the corpus.

LDA Implementation: Gensim [29], a library that provides a comprehensive set of tools for various topic modelling strategies and their evaluation, was used to implement LDA. We applied manual hyper-parameter optimization for the topic number parameter, and as a result it was selected to be 13, which yielded the best coherence score. For the alpha and beta parameters, we used the auto feature of Gensim, which lets the model learn the best values itself.

Book Inspections

In order to get a clearer view on the corpus features, we have collected the related meta data for each book including the author name, total edition count, publication date, target audience and reviewed the technical and practical aspects. Every book has explicitly stated the target audience and how the book can be used.

After the metadata collection, we determined the addressed topics related to software testing for each book from the chapters that we included in our corpus. We followed a labeling methodology, in which the labels were created from the topic names that were extracted from the literature we presented in Section 2. The following steps were established to analyze the per book topic coverage:

- First, we checked the table of contents sections, labeling each chapter based on the information present. While some of the books had a very detailed table of contents section, capturing the main and subtopics, a lot of the table of contents were more broad. Thus, the amount of topics saved for each book showed a big variation in this step.
- We went through every (main) chapter and added labels based on the topics of the subsections. In most of the cases, these topics were interpreted directly from the subsection titles. However, we also took the first and last paragraphs of each main section into account to enhance our interpretation of the topics since they give an overall idea on what the section is about, but also some subsection titles such as single word titles were not explanatory. When the subsection titles were not representative of a topic, we again checked the first and last paragraphs and any (sub) subsections to interpret the topic. It was common to have sub subsections. We only took the topics of such sections into account if they provided an insight that was not already covered by the parent sections.
- Some of the books had a summary section at the end of each chapter. If such a section is present, we checked the summary sections and added any other topic that was discussed in the chapter which was not captured by our previous labels.
- We organized the results in our topic coverage matrix, in which the topic names are organized as the columns and

the book names as the rows. A topic for a book is only marked if it had the corresponding label.

The testing tools and frameworks that were discussed in the books are extracted after the manual inspection of each chapter. For each chapter, we used helper search strings to target the tools, frameworks and suites that are mentioned within the chapter.

The extent and depth of the topics discussed are not reflected in our manual inspection results because our main aim with the manual inspection process is to see how often a topic is addressed throughout the corpus (even if a topic was discussed in a small section we labeled and marked it in our matrix). LDA, however, informs us on the probability distribution and significance of the captured topics.

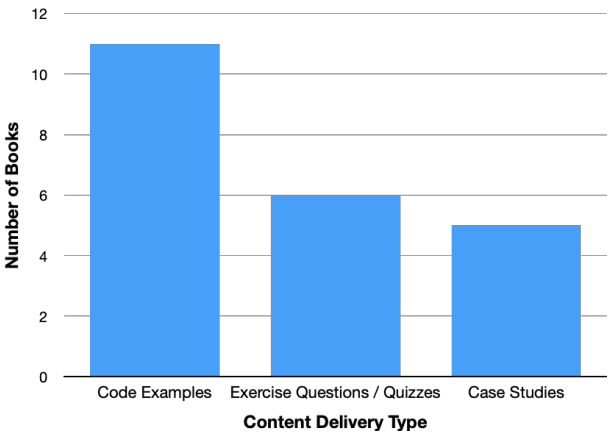

Figure 1: Content Delivery Types

4 Results

In this section we present the results we derived from the inspection on the books and the LDA topic modelling.

4.1 Corpus Characteristics

More than half of the popular books that we collected were published before 2010, with the oldest one being published in 1999 (Software Test Automation) and the newest ones in 2020 and 2019 (The Pragmatic Programmer and The Foundations of Software Testing respectively). In this regard, our corpus can be considered to be mostly composed of old books (only 4 of them were published after 2015). This shows that even though these books do not have the most updated information and trends about software testing, they are preferred for the concepts they discuss.

To assess the practicality of the books, we looked into how the technical information was supported. We found that example codes, case studies, exercises and quiz questions were the main ways of engaging the reader with more practical content.

The most popular method is the code examples. It is very common to support the discussion of a testing topic with an example test code related to a specific context. However, the example codes are not just limited to test code. Code snippets

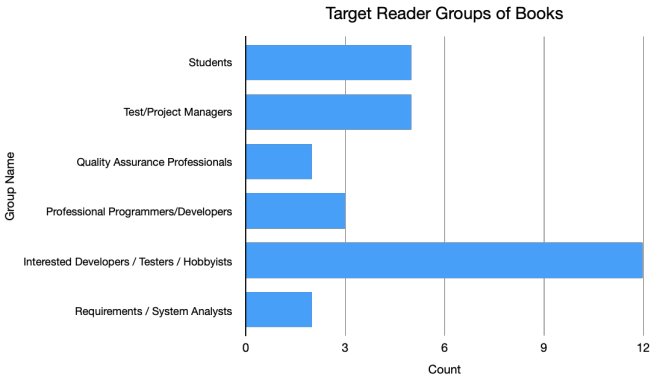

Figure 2: The occurrence count of target reader groups

of class and method designs to showcase concepts such as boundary analysis, unit testing, data flow, test design etc. are also a part of this content delivery type.

Secondly, almost half of the books included some sort of exercise and practice questions (most commonly at the end of a chapter). We observed that all the books that utilized this method also included example codes.

Lastly, more than a quarter of the books included case studies related to test management process of certain companies, and experiences of individuals as testers, test managers, developers etc.

We also analysed the target audience groups of the books to get a better insight of why the content is presented in a certain way. Extracting this information was straightforward, as every book either had a section dedicated to talk about the audience or included this information at the first sections such as preface or foreword. Most of the books had more than one specific target group. Almost all the books explicitly state that any tester, developer or hobbyist are amongst the target reader groups. The second most targeted groups are students and test managers.

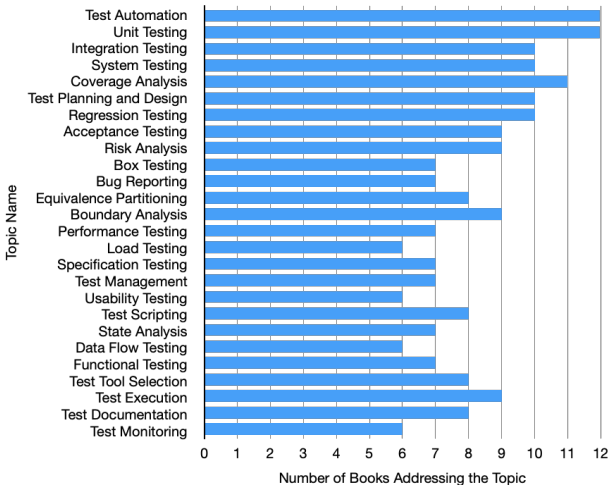

Figure 3: Top Topics From Coverage Matrix

Topic Name	Keywords
1 - Test Automation and Tools	0.019*"tool" + 0.010*"automation" + 0.009*"data" + 0.007*"problem" + 0.007*"system" + 0.006*"automated" + 0.006*"bug" + 0.006*"application"
2 - Agile Testing	0.012*"story" + 0.009*"product" + 0.009*"development" + 0.008*"agile" + 0.006*"system" + 0.006*"business" + 0.006*"release" + 0.006*"project"
3 - Coverage Analysis	0.019*"path" + 0.014*"coverage" + 0.013*"node" + 0.013*"graph" + 0.012*"set" + 0.012*"variable" + 0.010*"input" + 0.009*"criterion"
4 - Service Load Testing	0.016*"system" + 0.014*"service" + 0.014*"request" + 0.008*"task" + 0.007*"load" + 0.007*"client" + 0.007*"user" + 0.006*"server"
5 - Test Planning and Design	0.016*"plan" + 0.015*"design" + 0.011*"system" + 0.011*"project" + 0.010*"level" + 0.010*"process" + 0.009*"risk" + 0.009*"quality"
6 - Model Based Testing	0.023*"system" + 0.010*"event" + 0.008*"state" + 0.008*"error" + 0.007*"input" + 0.007*"level" + 0.007*"output" + 0.006*"model"
7 - Boundary and Equivalence Analysis	0.026*"class" + 0.010*"equivalence" + 0.008*"boundary" + 0.008*"object" + 0.006*"input" + 0.006*"net" + 0.005*"oriented" + 0.005*"system"
8 - Test Scripting	0.052*"script" + 0.023*"file" + 0.022*"comparison" + 0.016*"outcome" + 0.016*"data" + 0.014*"set" + 0.011*"testware" + 0.010*"execution"
9 - Testing Levels	0.022*"system" + 0.012*"integration" + 0.011*"unit" + 0.007*"call" + 0.007*"incident" + 0.006*"configuration" + 0.005*"device" + 0.004*"constituent"
10 - Bug Reporting	0.048*"bug" + 0.015*"report" + 0.010*"project" + 0.009*"programmer" + 0.007*"error" + 0.005*"fixed" + 0.005*"fix" + 0.004*"information"
11 - Specification Based Testing	0.007*"specification" + 0.007*"set" + 0.007*"boundary" + 0.006*"model" + 0.005*"element" + 0.004*"combination" + 0.004*"state" + 0.004*"function"
12 - State Analysis	0.014*"state" + 0.011*"model" + 0.005*"property" + 0.005*"execution" + 0.004*"lookup" + 0.004*"proc" + 0.004*"line" + 0.004*"finite"
13 - Box Testing	0.045*"module" + 0.013*"box" + 0.009*"white" + 0.006*"dynamic" + 0.005*"driver" + 0.004*"bottom" + 0.004*"black" + 0.004*"stub"

Table 2: Topics Captured by LDA

4.2 LDA Results

The LDA based topic modeling analysis implementation resulted in 13 topics. We used pyLDAvis [30], an interactive library for topic modelling visualizations, to visualize the topic clusters on the intertopic distance map. Each circle in the map represents a topic, where a larger circle means a higher dominance in the corpus and the distance between two topics shows their relevancy. We see that our topics are well divided and distributed across the map. The pyLDAvis visualization is depicted in Figure 4. Top 8 keywords with their corresponding weights for each topic are presented in Table 2. The topic labels were interpreted manually based on these 8 keywords for each topic.

Based on the LDA results, the most dominant topics in our corpus include topics like test automation and tools, agile testing, coverage analysis, service load testing, test planning and design. Out of the topics that are captured by LDA, white and black box testing was the least dominant one.

4.3 Topic Coverage Matrix Results

The LDA results give an overview of the common topic patterns within our corpus. To see the topic coverage information for each book specifically, we created a topic coverage matrix as a result of our manual inspection. Test automation and unit testing were addressed in 85% of the books. Secondly, almost every book discussed different test coverage analysis and metrics. Integration, system, regression, acceptance testing were the third most often covered topics. Test scripting, test tool selection, risk analysis, test execution, test documentation, test designs and plans were also mentioned in many books. Most frequent topics from our topic coverage matrix can be found in Figure 3. The least discussed topics include mobile application testing, web application testing, API testing, GUI testing, random testing, smoke testing, gray box testing and symbolic testing.

4.4 Combining LDA and Inspection Results

The differences between the LDA and manual inspection results occur, as LDA analyzes the corpus characteristics as a whole and shows us the dominance of topics, while our inspection results try to understand the semantics discussed per book and their frequency in the corpus.

We see that test automation is the most popular topic both in our LDA and coverage metric results meaning that it was the most frequently and extensively discussed topic.

Unit testing, which is also at the top of the list in our topic coverage matrix, is captured in the testing levels topic by LDA along with integration and system testing.

Although frequent topics, regression testing, acceptance testing, test execution topics were not captured by LDA.

Coverage analysis, the second most frequent topic in our matrix, is ranked 3rd by the LDA.

While agile testing and service load testing topics were

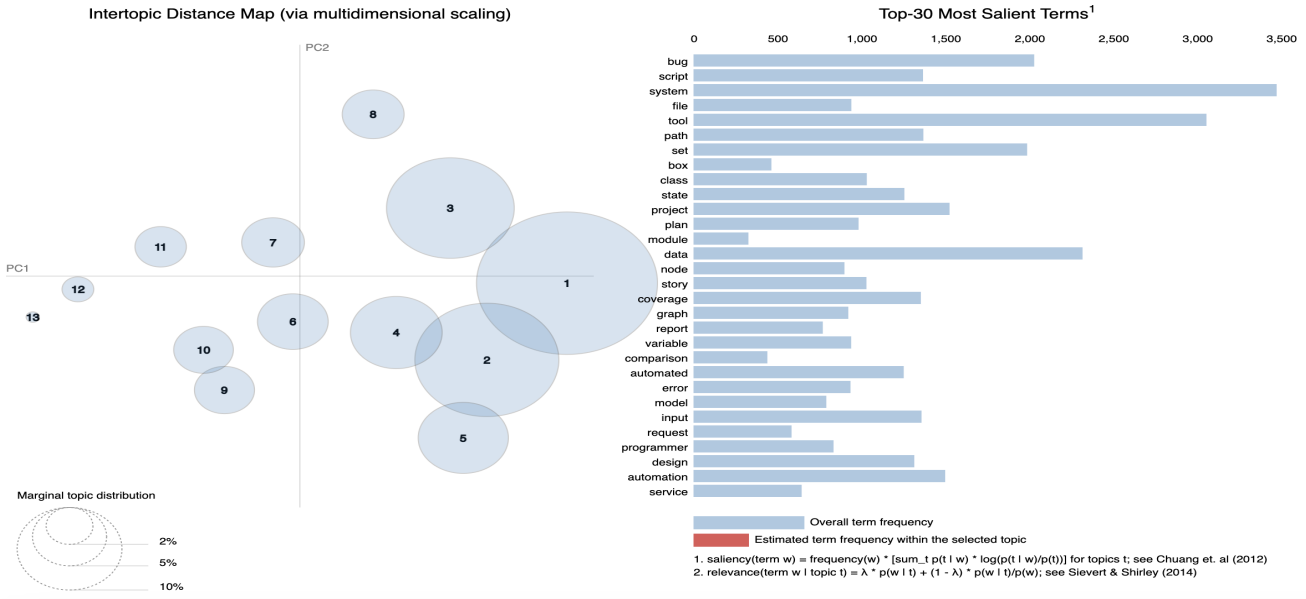

Figure 4: pyLDAvis Intertopic Distance Map

ranked highly by the LDA, they were not very frequent in our corpus (addressed by less than half of the books). We believe the main reason for this is that these topics has dedicated books (Agile Testing, System Reliability Engineering), which extensively address these topics and take a significant part of our corpus.

Another topic that was captured by LDA, but was not very frequent is model based testing.

Test planning and design topic was ranked highly by LDA and our matrix.

Boundary analysis, equivalence partitioning, test scripting, state analysis are also frequent topics that are captured by LDA.

Test execution and documentation topics were discussed by more than half of the books but were not captured by LDA.

Based on our results we see that topics such as test automation, test design and test planning are common and extensively discussed topics which are also highly demanded by the industry.

None of the books addressed the intelligence testing area and the representation of trending topics such as web and mobile application testing was very small.

4.5 Most Discussed Frameworks and Tools

It was interesting to see that some of the books did not discuss or mention any software testing frameworks or tools. Unit testing frameworks (xUnit) were the most addressed frameworks with JUnit being the most popular one, more than half of the books giving it a place. There were not many other frameworks or tools that were covered by the same books. Selenium, and Purify (a dynamic memory analysis tool) were mentioned in 2 books each.

Overall, none of the books explicitly had a focus on the practical usage of specific testing frameworks or tools. However, testing tool selection is a popular topic as our topic analysis results show. This topic mainly focused on concepts such as the criteria that needs to be considered before choosing testing tools, validating the results of testing tools, being aware that the testing tools can be faulty.

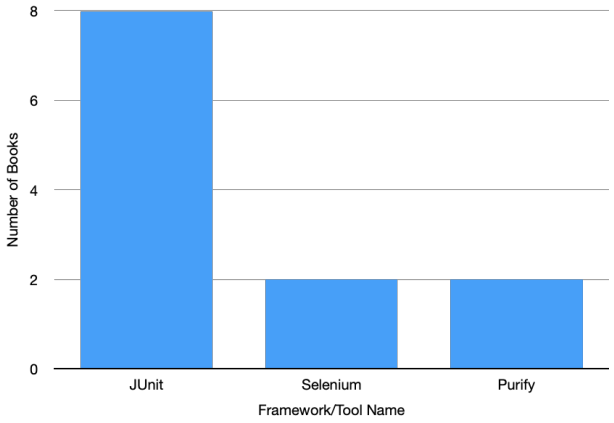

Figure 5: Top Frameworks and Tools

5 Responsible Research

While conducting this research, we made sure that throughout the data collection, processing and analysis processes we adhered to the responsible research principles of reproducability and transparency.

For the book collection process, we clearly outlined the heuristics and criterion that we determined, which define what a popular software testing book is. In our replicaiton package, the heuristics that each book satisfied can be found.

As we discussed, we filtered out certain chapters of the books based on the criteria that they included some sort of

technical information regarding software testing or the hard skills of software testers. The information on which chapters are selected is outlined in our replication package.

Our raw manual inspection results, topic coverage matrix, the stop words that we picked and the hyper parameter information used in the LDA implementation, the html file generated by the pyLDAvis library are all documented and can be found in our replication package.

All the steps and procedures we followed are documented transparently and none of the results were manipulated, making our research reproducable.

6 Discussions and Limitations

In this section we outline our discussions about our data, results and limitations in our methodology.

6.1 Corpus

Each of our heuristics aim to target the popular books from different aspects in software testing field. After the addition of 9 books, which satisfied more than one heuristic to our data set, we saw that only 3 of those met the Amazon best sellers heuristic. To increase the weight of the Amazon heuristic, we added 3 books, which were ranked very highly by many users in the software testing category. Out of those 3, the books "The Pragmatic Programmer" and "Clean Code" were by far the more popular ones in terms of the rating and number of users rated. Although these books are not directly related to software testing, we still chose to include them in our data set because they represent the preferences of a large audience group in software testing field. However, the number of selected chapters from those books were very small compared to other books. Thus, although we wanted to increase the weight of the Amazon heuristic in our corpus by selecting the top ranked books, chapter wise these books had trivial effect in terms of weight.

6.2 Topic Results

We followed two topic analysis techniques to identify the major topics being discussed in the popular software testing books. While LDA showed the dominant topics of our corpus, our topic coverage matrix depicted the frequency of topics. We see that many of the frequent topics (test automation, unit testing, coverage analysis, test planning, boundary & equivalence analysis, test scripting, bug reporting, specification based testing, state analysis, box testing) were also captured by LDA. We conclude that these topics were discussed more extensively within our corpus compared to some other frequent topics (regression testing, acceptance testing, test execution, test documentation, performance testing, functional testing, test management, usability testing, data flow analysis), which were not captured by LDA. We observed that books mostly addressed the regression and acceptance testing topics within the discussions of testing levels.

There were also topics that were ranked highly by LDA, which were only discussed in less than half of the books (agile testing, load testing, model based testing). The topics agile testing and load testing had books dedicated to them, which had many chapters that took a significant portion of our corpus. Since LDA is a probabilistic model, the extent to which these topics were discussed in our corpus causes them to be ranked highly by LDA. The "model based testing" topic was explicitly mentioned in only 3 books. However, testing based on various model systems such as data flow and state machine models was a common subject, as it is also reflected in our topic analysis results.

We consider topics such as performance, load and security testing to be more specialised topics. We observed that at certain books these topics were given more depth. Compared to those topics, we expected test management and usability testing topics to be represented more in our corpus. The sections that were labeled as test management or usability testing were not very comprehensive and we did not observe these topics to be discussed more in depth at certain books. Considering that test managers were amongst the second most targeted audience groups and we have many books that focus on general testing practices in our corpus, these topics are not addressed to the level of extend we anticipated.

6.3 Method Limitations

We identified one main limitation on the application of LDA on a corpus of book chapters. Many of the books had figures, images, diagrams etc. to enhance the discussions on a topic. Such semantics, which were not presented in a textual context, could not be taken into account by LDA. This issue potentially causes some topics to be underrepresented by LDA (e.g., if a book utilized a lot of non-textual discussion for a topic). However, even if a topic was largely discussed through images or diagrams, we observed that they will still be supported or summarised by text. Thus, overall we estimate this effect does not have a significant impact in our results.

We used the default LDA model of the Gensim library in our LDA modelling, which uses a Bayesian model [28] [29]. There are alternative LDA models such as LDA Mallet [31], which utilize Gibbs sampling and may produce more precise results.

Our manual inspection method also has certain limitations. We labeled each main chapter based on their titles, topics of the sub sections and the first and last paragraphs. Although, the topics of the subsections were mostly extracted from their titles, at some parts these titles were not explanatory. In such cases we again checked the first and last paragraphs and the subsections to enhance our understanding. However, with this method we potentially miss some of the semantics and topics that are discussed within chapters, which do not have dedicated subsections.

7 Conclusions

In this section we readdress our research questions, summarize our conclusions and provide an insight for the possible future work.

7.1 Research Questions Revisited

RQ1: What are some software testing topics that are commonly discussed in the popular software testing books? Are there any patterns in terms of coverage of topics?
Test automation, coverage analysis, test design and planning,

testing levels (unit, integration, system) are the most frequent and extensively discussed topics in our corpus of popular books. Regression testing, acceptance testing, test execution, risk analysis are also amongst commonly discussed topics in the popular software testing books. These topics were covered through the utilization of various practical methods, code examples being the most common one.

RQ2: How do the topics covered in popular software testing books compare to the topics that are most relevant to the industry needs and rising software testing trends?

Industry needs: As previous work on the industry demands from software testers suggest [5][6][7][8], test automation is a highly demanded skill by the job recruiters with automation specialist being a seeked role. Test design and planning, performance testing, acceptance testing, user experience testing, test management, execution monitoring are other demanded areas from software testers. Based on our results, we see that popular software testing books address test automation, test design and planning topics comprehensively and can provide the readers with necessary skills that are expected from the industry. Although not as in depth acceptance testing was a commonly addressed topic. Test execution topic was addressed by many books, however test monitoring was covered by less than half of the books. Performance and test management topics were only mentioned by half of the books.

Trending Areas: Incorporation of AI and machine learning into the software testing cycle, predictive testing, web application testing, mobile application testing, automated test generation are some of the highly trending and accelerating topics in the software testing area [2] [3] [4]. As we discussed, a majority of the books in our corpus are old, meaning they do not cover the recent trending practices or topics. It was not unexpected to see that none of the books addressed the intelligent testing area. Only a few books covered the mobile and web application texting topics. Overall, our corpus of books mostly cover the essentials and foundations of software testing.

RQ3: Are there any major areas of software testing that are underrepresented in popular software testing books?

Although our corpus have some more specialised books, it also includes many books that are general software testing books. Thus, we expected topics such test management and usability testing to be addressed more extensively and frequently in our corpus. Test management topic was mentioned by half of the books and was not captured by the LDA model. Considering that test managers were among the second most targeted audience groups in our corpus, test management was not discussed as extensive as some other topics. Usability testing was only discussed in less than half of the books and was not identified by LDA. We observed that overall the sections dedicated to usability testing and test management were not comprehensive. Web and mobile application testing were very rare topics. Although load testing and agile testing topics were ranked highly by LDA, they were addressed in less than half of the books. However, we believe the core reason for this is that they are more specialised topics being discussed in books that have such focus areas. We consider performance and security testing topics to be other specialised topics. They are addressed by around half of the books but were not identified by LDA, being discussed in more depth at certain books. We conclude that besides web and mobile application testing, test management and usability testing topics are the major topics that are underrepresented in our corpus of books.

RQ4: What are the most common testing tools and frameworks discussed in popular software testing books?

JUnit is the most popular framework which is discussed by almost all the books. Selenium and Purify are second on the list, however they are addressed by only 2 books each. Overall, the popular software testing books do not focus on the application of specific tools, but on the tool selection process.

7.2 Implications and Future Work

Our study implies that the popular software testing books cover some major topics that are in line with the industry expectations from software testers. However, there are also such areas that needs more attention. Overall, the popular software testing books are not a good resource for individuals who want to learn about software testing trends or gain practical knowledge in specific testing frameworks and tools.

In the future this study can be extended by the addition of more books. The heuristics to select a popular book can be made limited to more recent books in order to get a better understanding on the topic coverage of the new popular books. Working with only the recent books can provide more insights in the discussions of the trending topics.

References

[1] Vahid Garousi, Austen Rainer, Per Lauvås, and Andrea Arcuri. Software-testing education: A systematic literature mapping. *Journal of Systems and Software*, 165:110570, 2020.

[2] Fatih Gurcan, Gonca Gokce Dalveren, Nergiz Ercil Cagiltay, Dumitru Roman, and Ahmet Soylu. Evolution of software testing strategies and trends: Semantic content analysis of software research corpus of the last 40 years. *IEEE Access*, 10:106093–106109, 2022.

[3] Alireza Salahirad, Gregory Gay, and Ehsan Mohammadi. Mapping the structure and evolution of software testing research over the past three decades. *Journal of Systems and Software*, 195:111518, 2023.

[4] Mohamed Boukhlif, Mohamed Hanine, and Nassim Kharmoum. A decade of intelligent software testing research: A bibliometric analysis. *Electronics*, 12(9), 2023.

[5] Raluca Florea and Viktoria Stray. The skills that employers look for in software testers. *Software Quality Journal*, 27(4):1449–1479, 2019.

[6] Raluca Florea and Viktoria Stray. A global view on the hard skills and testing tools in software testing. pages 143–151, 2019.

[7] Raluca Florea, Viktoria Stray, and Dag I.K. Sjøberg. On the roles of software testers: An exploratory study. *Journal of Systems and Software*, 204:111742, 2023.

[8] Mohamad Kassab, Phillip Laplante, Joanna Defranco, Valdemar Vicente Graciano Neto, and Giuseppe Destefanis. Exploring the profiles of software testing jobs in the united states. *IEEE Access*, 9:68905–68916, 2021.

[9] Dorothy Graham, Rex Black, and Erik van Veenendaal. *Foundations of Software Testing: ISTQB certification 4th Edition*. Cengage Learning, EMEA, 2019.

[10] Cem Kaner and James Bach. *Lessons learned in software testing*. Wiley, 2001.

[11] Myers, Corey Sandler, and Tom Badgett. *The art of software testing 3rd Edition*. John Wiley & Sons, 2011.

[12] Lisa Crispin and Janet Gregory. *Agile testing a practical guide for Testers and agile teams*. Addison-Wesley, 2008.

[13] Paul Ammann and Jeff Offutt. *Introduction to software testing*. Cambridge University Press, 2016.

[14] Paul Jorgensen. *Software testing: A craftsman's approach*. Tectum Verlag, 2013.

[15] Stefan P. Jaskiel and Rick D. Craig. *Systematic software testing (Artech House Computer Library)*. Artech House, 2002.

[16] David Thomas and Andrew Hunt. *The pragmatic programmer your journey to mastery: 20th anniversary edition*. Addison-Wesley, 2020.

[17] Robert Cecil Martin. *Clean code: A Handbook of Agile Software Craftsmanship*. Prentice Hall, 2009.

[18] Betsy Beyer, Chris Jones, Jennifer Petoff, and Niall Murphy. *Site Reliability Engineering*. O'Reilly Media, 2016.

[19] Lee Copeland. *A practicioner's guide to software test design*. Artech House Publihsers, 2004.

[20] Mauro Pezze and Michal Young. *Software testing and analysis: Process, principles, and Techniques*. John Wiley & Sons, 2007.

[21] Ron Patton. *Software testing 2nd Edition*. Sams Publishing, 2005.

[22] Mark Fewster and Dorothy Graham. *Software test automation effective use of test execution tools*. Addison-Wesley, 1999.

[23] Vahid Garousi, Austen Rainer, Per Lauvås, and Andrea Arcuri. Software-testing education: A systematic literature mapping. *Journal of Systems and Software*, 165:110570, 2020.

[24] Andy Zaidman and Baris Ardic. 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)*, pages 9–16.

[25] Istqb®. https://www.istqb.org/, May 2023.

[26] Amazon best sellers: Best software testing. https://www.amazon.com/Best-Sellers-Software-Testing/zgbs/books/4133, May 2023.

[27] Doruk Ozmetin. Replication Pacakge for Topic Analysis on Popular Software Testing Books. https://github.com/dozmetin/st_book_analysis, 12 2023.

[28] David M. Blei, Andrew Y. Ng, and Michael I. Jordan. Latent dirichlet allocation. *Journal of Machine Learning Research*, 3(4-5):993 – 1022, 2003.

[29] Radim Rehurek. Topic modelling for humans. https://radimrehurek.com/gensim/, 2022.

[30] Pypi. https://pypi.org/project/pyLDAvis, 2023.

[31] Latent dirichlet allocation via mallet. https://radimrehurek.com/gensim_3.8.3/models/wrappers/ldamallet.html, 2023.