An Empirical Study on Continuous Integration Trends, Topics and Challenges in Stack Overflow

Ali Ouni
ETS Montreal, University of Quebec
Montreal, QC, Canada
ali.ouni@etsmtl.ca

Islem Saidani
ETS Montreal, University of Quebec
Montreal, QC, Canada
islem.saidani@ens.etsmtl.ca

Eman Alomar
Stevens Institute of Technology
Hoboken, New Jersey, USA
ealomar@stevens.edu

Mohamed Wiem Mkaouer
Rochester Institute of Technology
Rochester, New York, USA
mwmvse@rit.edu

ABSTRACT

During the last few years, Continuous Integration (CI) has become a common practice in open-source and industrial environments to reduce the scope for errors and increase the speed to market through the automated build and test processes. However, despite this wide adoption throughout the years, little is known about the challenges developers discuss. Analyzing the discussions of developers is required to understand what researchers, educators and practitioners should focus on, and how discussion communities can be helpful to shed the light on CI challenges. In this study, we examine Stack Overflow (SO), the most popular crowd-sourced forum, to understand the challenges developers face in the CI context. We collect a corpus of 27,728 CI-related developers posts from SO and analyze those posts through a mixed method with quantitative and qualitative analyzes. To study the trends of CI discussions, we investigated the metadata of CI questions, users and tags. Then, we extract the CI main topics using Latent Dirichlet Allocation (LDA) tuned with Genetic Algorithm (GA). Finally, we investigate the most popular and difficult topics faced by developers based on unanswered questions to get further insights into CI challenges. The LDA clustering reveals that developers face challenges with six main topics namely Build, Testing, Version Control, Configuration, Deployment, and CI Culture. Particularly, we found that the build topic is the most popular among the studied topics and that version control and testing topics are the most difficult for the SO community. Our study uncovers insights about CI challenges and adds evidence to existing knowledge about CI issues related especially to software build.

ACM Reference Format:
Ali Ouni, Islem Saidani, Eman Alomar, and Mohamed Wiem Mkaouer. 2023. An Empirical Study on Continuous Integration Trends, Topics and Challenges in Stack Overflow. In *Proceedings of the International Conference on Evaluation and Assessment in Software Engineering (EASE '23), June 14–16, 2023, Oulu, Finland.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3593434.3593485

1 INTRODUCTION

Continuous Integration (CI), a common software engineering practice that is widely-adopted in industry and open-source environments [33]. CI advocates continuously integrating code changes, by automating the process of build and testing [12], which reduces the cost and risk of delivering defective changes. Nevertheless, introducing changes under such a context is still risky and can lead to productivity loss [8], release delays [36], and cost impacts [20].

Prior studies that have examined CI challenges relied mainly on surveys/interviews of a selected number of stakeholders [10, 17, 24]. Although interviewing developers provides great insight, it has a major limitation: it cannot be generalized due to the limited number of interviewees. Therefore, there is a need to investigate CI challenges on a large scale. At the same time, we observe that discussions related to CI are becoming increasingly prevalent in online developer forums, to find answers to their CI related issues. Stack Overflow (SO)[1] is one of the most popular Q&A sites for developers, by recording over 19 million questions in 2020 [22]. For instance, in one of the SO posts[2], a developer asked: 💬*"I try to setup CI for Go app and Jenkins.. So, my questions are (1) If my app will contain much more code file, packages etc. should I still build it as go build main.go? (2) How correctly give name for go build output file to add it to the artifacts? (3) Should I use some kind of make file/script etc to collect dependencies on build machine? What is best practice here?"*. This question received an answer only after over four years despite being viewed over 1k times. Which may indicate that the issue faced by this developer is difficult to resolve.

We, therefore, believe that an analysis of CI discussions on SO can help the research community and CI stakeholders in better understanding developers' concerns and hence improving the adoption of CI. To this aim, we conduct in this paper a large-scale empirical study of 27,728 CI related posts on SO to address the following Research Questions (RQs):

RQ1: (Trends) How have CI discussions grown since the creation of Stack Overflow? We aim to gain insights into the temporal trends of CI discussions. Specifically, we study the volume of questions (11,641) and their answers (16,087), the users responsible

[1]https://stackoverflow.com/
[2]https://stackoverflow.com/questions/34731416

for creating such posts (17,992) and also the tags (2,806) associated with CI questions. Results show that developers frequently use SO to seek help with their CI problems. We also found that SO users are showing more interest in CI over the years and that most CI questions tags are around CI servers and platforms.

RQ2: (Topics) What topics are discussed around CI? We leverage Latent Dirichlet Allocation (LDA) technique to identify the key topics that developers discuss on SO. Further, we use an advanced parameter tuning technique based on Genetic Algorithm (GA) to find the optimal parameters of the LDA algorithm. Our tuned LDA reveals that CI discussions cover six main topics namely *Build, Testing, Version Control, Configuration, Deployment* and *CI Culture*. The primary driver behind these questions is to enhance the usage of CI tools/infrastructures in the development process. Specifically, around 40% of discussions are related to build process suggesting this phase is a key concern within CI projects.

RQ3: (Challenges) Which topics are the most popular and difficult among CI related questions? We exploit the information provided in SO to discover the topics being the most popular and difficult to answer CI questions. In addition, we examine a significant simple of unanswered questions to gain further insights into the CI challenges. Our findings reveal that build related questions are the most popular (in terms of view, favorite and score questions). Additionally, questions related to version control involve the highest rates of unanswered questions, while testing questions require the longest time to receive accepted answers. When examining the unanswered questions, we found that those questions usually receive responses in the form of comments where users either suggest solutions to address the problem or ask for more clarification.

The main contributions of this study are summarized as follows:

- We conduct the first empirical study to mine and extract 27,728 SO posts related to CI to better understand the challenges, trends and topics around CI.
- We perform a mixed-method study, through quantitative and qualitative analyses, to shed light on characteristics of CI related topics and the usage of CI tools/infrastructures.
- We provide practical implications of our findings for researchers, developers, tool builders and educators.
- We publicly provide a replication package [1] that contains the dataset and scripts in order to replicate our results.

The remainder of this paper is organized as follows. In Section 2, we review the most related works to our study. In Section 3, we describe our research methodology and reveal the main findings of our study. In Section 4, we discuss the implications of our work. Then, we review threats to validity in Section 5, and finally we address the conclusions to draw in Section 6.

2 RELATED WORK

CI practice. Many research efforts have outlined the outcomes of CI adoption. For instance, Vasilescu et al. [33] and Hilton et al. [18] studies reveal a significant improvement in the team's productivity in terms of the frequency of Pull Requests (PRs). Zhao et al. [42] evaluated the effects of CI adoptions on some development practices, such as code writing and submission. They found that CI practice aligns with CI principals such as the "commit often" and "merging small commits" guidelines recommended by Fowler [12].

On the other hand, other works did not observe these outcomes in their studies. For example, Bernardo et al. [8] have observed that CI does not always reduce the time for delivering the changes. Rahman et al. [25] have observed for the studied Open Source Software (OSS) projects some CI benefits such as improvements in bug and issue resolution. However, for the proprietary projects, they did not observe such benefits. Recently Saidani et al. [29] investigated the impacts of CI on code refactoring practice. They found that the adoption of CI is associated with a drop in the refactoring size as recommended, while refactoring frequency as well as the number of developers that perform refactoring are estimated to decrease after the shift to CI, indicating that refactoring is less likely to be applied in CI context.

CI challenges Despite its valuable benefits, CI adoption brings with it many challenges. In the following, we review the most relevant papers that discuss CI challenges and bad practices. It is worth pointing out that we selected these works through a hybrid process: (i) performing a query in Google Scholar[3] and Scopus[4] and (ii) snowballing by searching in the cited papers of the related work previous SLR about CI challenges.

Debbiche et al. [10] interviewed 11 developers at a major communication company to investigate what challenges they faced when adopting CI. Their case study revealed a list of challenges including testing, code dependencies and tools and infrastructures. Laukkanen et al. [20] interviewed 27 stakeholders at Erickson and found that the main challenges are due to the lack of time, unstable tests, tools and team organization. Zahedi et al. [40] have exploring continuous software engineering (CSE) from the practitioners perspective by mining discussions from Q&A websites. They found that number of discussions related to CSE has been increasing sharply, and among the most discussed tools of CSE, free and/or open source tools were highly popular. Shahin et al. [31] used Systematic Literature Review (SLR) method for reviewing the peer-reviewed papers on continuous practices published between 2004 and 2016. Their findings revealed a list of critical factors including testing effort and time, team awareness and transparency, good design principles and appropriate infrastructures that should be carefully considered when introducing continuous practices in a given organization. Another SLR was performed by Laukkanen et al. [19] who identified forty problems related to CI/CD adoption. The most critical reported problems were related to testing, merging conflicts and system design. Hilton et al. [17] have found, by interviewing 51 developers, that the latter face many trade-offs between speed and assurance, between better access and information security, and between more CI configuration options and better flexibility in use. Additionally, the main challenges they face are related to build failure and time, lack of tool support and the setting of CI servers. Pinto et al. [24] surveyed 158 CI users in order to investigate work practices and challenges in CI. Their main results reveal that CI main challenges are (i) the build environment, (ii) inadequate testing and time pressure and (iii) false sense of confidence in CI servers' feedback. Widder et al. [36] conducted a mixed

[3]https://scholar.google.com/
[4]https://www.scopus.com/

method, including a survey and statistical modeling from 6,239 projects to identify the reasons behind CI abandonment. There results revealed that many developers find that the most difficult issues to resolve are (1) build failures, (2) complex tool setups like Docker and (3) long build times. Vassallo et al. [34] survey with 124 developers revealed four relevant anti-patterns (*i.e.,* practices that contradict CI principles) of CI namely slow build, skipping failed tests, broken release branch and late merging. Elazhary et al. [11] found, by performing a case study and activity log mining, that CI main challenges are related to builds, dependency management and testing. Zampetti et al. [41] also investigated CI anti-patterns by performing a mixed method consisting of interviewing 13 experts and 26 developers and mining of 2,300 SO posts. Specifically, they identified 79 CI anti-patterns practiced by developers related to different dimensions of a CI pipeline including infrastructure, build process and quality assurance. Our study is different from Zampetti et al. [41] study as we focus on the CI challenges developers face while they investigate the bad practices of developers that are not conform to CI principles. Moreover, surveys/interviews might represent a limited resource of information and therefore cannot be generalized. Further, there are other sources of information for examining CI challenges and SO is one of them. This Q&A website is a popular venue for developers who seek advice to resolve many technical problems/issues. We fill this gap with our empirical study, where we analyze developers discussions to investigate CI trends, topics and challenges.

3 EMPIRICAL STUDY DESIGN & RESULTS

The main *goal* of this study is to obtain and share insights with CI stakeholders regarding how CI is discussed in practice by analyzing Stack Overflow (SO) posts. Figure 1 depicts an overview of our study design. Our methodology comprises two main steps: (*1*) CI posts extraction and (*2*) analysis method. In the following, we present the details of each of these two steps.

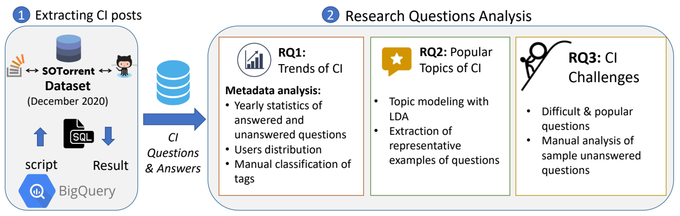
1 Extracting CI posts
SOTorrent Dataset
(December 2020)
script
SQL
Result
BigQuery
CI Questions & Answers
2 Research Questions Analysis
RQ1: Trends of CI
Metadata analysis:
• Yearly statistics of answered and unanswered questions
• Users distribution
• Manual classification of tags
RQ2: Popular Topics of CI
• Topic modeling with LDA
• Extraction of representative examples of questions
RQ3: CI Challenges
• Difficult & popular questions
• Manual analysis of sample unanswered questions

Figure 1: Overview of our study methodology

3.1 Extracting CI posts

3.1.1 SOTorrent Database. In this study, we mine CI questions & answers based on the SOTorrent dataset [5]. We particularly utilized the latest dump, dated on December 2020, available on Google BigQuery[5] that allows executing SQL queries on various public datasets. The dataset contains information about questions, answers and their metadata such as the creation date, score and view count.

[5]https://cloud.google.com/bigquery

3.1.2 CI posts. To extract relevant discussions on SO, we query questions from the SOTorrent dataset being tagged with *"continuous integration"* or that contain this term in their title texts. Similarly to Peruma et al. [23], we excluded searching for the term in the body of the post to avoid the increase of false positives in our dataset. Table 1 summarises the collected data. As shown in the table, we extracted 11,641 CI related questions, from which 1,981 (17.1%) questions did not receive an answer, 5,382 (46.2%) had an accepted answer, while 4,278 (36.7%) questions received at least one, but not accepted, answer.

Table 1: Statistics about the collected data.

Item	Count
Number of posts	27,728
Number of questions	11,641
Number of answered questions	9,660
Number of accepted answers	5,382
Number of distinct tags	2,806
Number of distinct users	17,992
Average number of tags per question	4
Average number of answers per question	1.4

3.2 Research Questions Analysis & Results

In this paper, we address three Research Questions (RQs). As described in Section 1, RQ1 aims to examine the temporal trends and growth of CI posts, users as well as the tags used to describe the questions. In RQ2, we classify the CI discussions into topics based on Natural Language Processing (NLP) techniques. Then, in RQ3, we based on the results of RQ2 to identify the most popular and challenging topics being discussed among developers. In the next subsections, we explain the rationale behind each RQ and our approach to to answer it. Then, we present our main findings.

3.2.1 RQ1. (Trends) How have CI discussions grown since the creation of SO?. **Approach.** First, we explore the volume of CI posts (*i.e.,* questions and answers) Then, we analyse the developers' involvement in CI discussions and whether a subset of SO users are responsible for the majority of CI related questions and answers. Then, to determine the concepts and technologies being associated with CI questions, we manually review the tags and classify them in order to determine the different categories these tags fall under.

Results. *Trends of Questions.* Figure 2 shows the yearly growth of CI questions with and without an accepted answer. The red bars represent the questions without an accepted answer, while the blue bars are questions with an accepted answer. The total number of questions is represented by the grey bars. Similarly to Peruma et al. [23], we eliminate SO data from the years 2008 (SO was launched this year) and 2020 as they contain incomplete information.

A first look at these bar plots shows that, as the years pass, developers are showing more interest in CI since the number of total questions increases each year. The only exception is for the year 2014 in which the number of questions decreased by 36 as compared to 2013. Regarding questions with an accepted answer, except for the years 2013, 2014 and 2018, we found that the related number of posts is also growing; while the number of questions without accepted answers is always increasing. Additionally, from the year 2014, the number of questions without accepted answers outnumbered questions with an accepted answers.

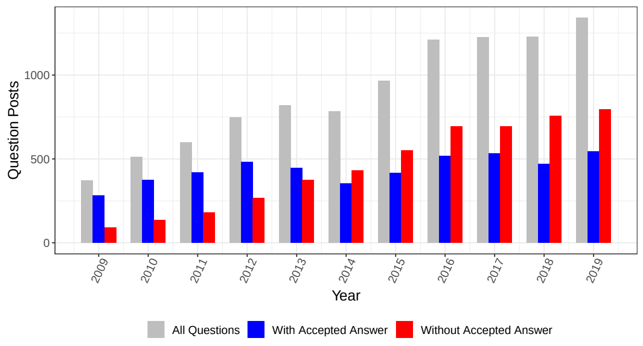

Figure 2: Count of CI questions and their answers per year.

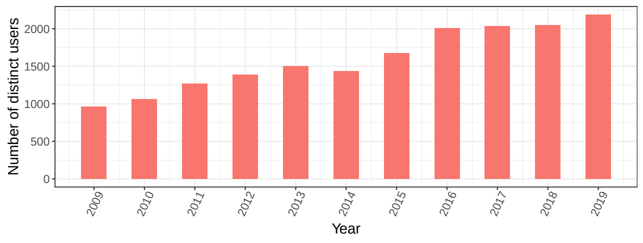

Figure 3: Distinct users involved in CI discussions per year.

Trends of Users. Next, we look at the involvement of the SO community members in CI discussions. We investigate unique users who post questions and answers and analyze their trends over the years. Figure 3 reveals an increasing trend in the number of unique users involved in CI discussions, except for the year 2014 (similarly to the number of CI questions shown in Figure 2). Hence, we believe that the dip in questions and users in 2014 is an interesting phenomenon that would require further investigation to explain the reasons behind it.

Next, we investigate the distribution of unique users for both questions and answers. As shown in Table 2, we see that 9,566 distinct users created 11,641 CI related questions (Table 1). As for answers, we observe that 4,525 distinct users posted accepted answers, while 5,589 distinct users are responsible for non-accepted answers. We also see that 87.36% of the users asked at most one question; which is applied to accepted and non-accepted answers. Indeed, 88% and 92% of the users were associated with only one accepted and non-accepted answer respectively.

Trends of Tags. As a further step to study the trends of CI discussions, we investigate the categories under which fall the tags associated with CI-related questions. In total, our dataset contains 2,806 distinct tags excluding *"continuous integration'"* tag. The top-10 tags are all related to servers (jenkins 7.9%, teamcity 2.3%, azure-devops 2.1%, hudson 1.6%), platforms (docker 1.7%, gitlab 2.3%, github 1.4%), tools (git 2.7%), programming languages (Java 1.5%) and concepts (continuous-deployment 3.12%). These popular tags represent 27.02% of the tags in the dataset as shown in Table 3.

Next, we plot the yearly growth of these popular tags as shown in Figure 4. Recall that we ignore the years 2008 and 2020 due to their incomplete data. Regarding CI servers, we observe that, except for "azure-devops", all the related posts are decreasing from a particular year. For instance, we see that the volume of "jenkins" tag shrinks from the year 2017 while the number of "hudson" tagged posts show a steep decline from 2012. Additionally, the popularity of posts

Table 2: Distribution of the number of posts created by a user.

Number of posts created by a user	count	Percentage
Questions -total distinct users	*9,566*	
1	8,357	87.36%
2	844	8.82%
3	195	2.04%
4	94	0.98%
5	32	0.33%
Others	44	0.46%
Accepted Answers - total distinct user:	*4,525*	
1	3,996	88.31%
2	376	8.31%
3	93	2.06%
4	35	0.77%
5	10	0.22%
Others	15	0.33%
Non-Accepted Answers - total distinct user:	*5,589*	
1	5,154	92.22%
2	336	6.01%
3	64	1.15%
4	15	0.27%
5	8	0.14%
Others	12	0.21%

Table 3: Top-10 most popular tags.

Tag	Occurrence	Percentage
jenkins	2,647	7.98%
continuous-deployment	1,035	3.12%
git	899	2.71%
teamcity	771	2.32%
gitlab	770	2.32%
azure-devops	697	2.10%
docker	589	1.77%
hudson	534	1.61%
java	528	1.59%
github	490	1.48%
Others	24,210	72.98%

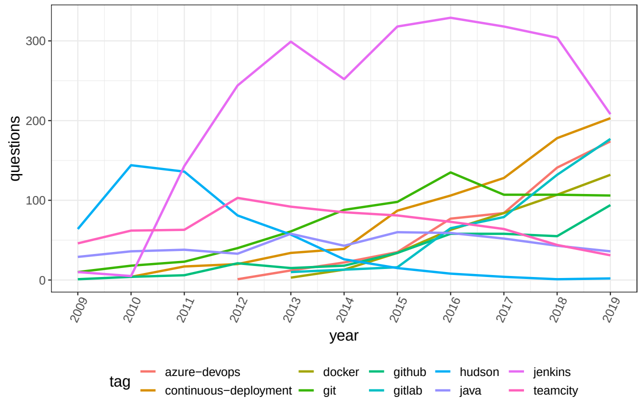

Figure 4: Yearly growth of popular tags associated to CI posts.

related to "github", "gitlab", "docker" and "continuous-deployment" seems to be in constant increase over the years. We also see that the curve of "git" and "java" tags are relatively constant.

Finally, we manually classify the tags to determine the most popular technologies/concepts used in CI discussions. To this aim, we manually reviewed a statistically significant sample of tags composed of 538 tags which represents a confidence level and interval of 99% and 5% respectively. As a result, we clustered the tags into seven categories namely *Platforms/Servers*, *Tools/IDEs*, *Programming Languages*, *Framework/Library/API*, *Concepts*, *Operating Systems*, and Other. Figure 5 shows the distribution of the instances associated with each tag category. Most of the tags (40.15%) are related to platforms and servers which include CI servers (*e.g.,* Jenkins, Teamcity) and platform products such as Dockerand Amazon Web Services. The next most popular category is related to general concepts including CI principals (*e.g.,* continuous deployment, continuous delivery and build automation), repository (*e.g.,* pull request, versioning) and programming (*e.g.,* variables, command-line). The third most popular tags fall under the Tools/IDEs category (17.85%). In this category, we found that tools are related to version control (*e.g.,* Git), build (*e.g.,* Maven, MSbuild), testing (*e.g.,* Selenium, NUnit), configuration (*e.g.,* chef-recipe) and code analysis (*e.g.,* FindBugsand Phpcs). The tagged posts include popular IDEs such as Xcodeand Visual Studio. Tags related to programming languages represent only 8.5% of the tags in our sample set. In this category, we found that the top-3 most popular languages are Java, python and C#. The category of frameworks/libraries/APIs which represents 5.7% of the tags include popular frameworks such as Node.js, Ruby on Railsand Angular. With regards to Operating Systems (OS) category, we found that the popular tags are mobile-based (*e.g.,* Android and iOS).

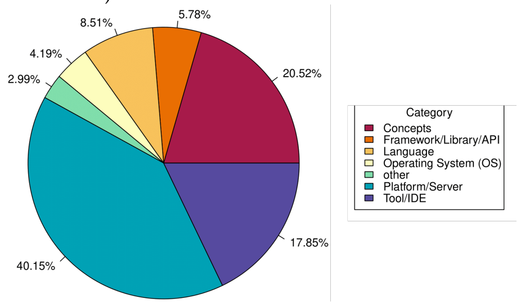

Figure 5: Distribution of the tags' categories.

3.2.2 *RQ2. (Topics) What topics are discussed around CI?.*

Approach. To explore the high level issues facing CI developers, we use Latent Dirichlet Allocation (LDA) technique [9] to aggregate and discover what is being asked in the CI posts. LDA has been recognized as one of the best unsupervised Machine Learning (ML) techniques that has shown its effectiveness in clustering a large volume of text documents [22, 39]. In order to cluster CI posts, we build a corpus in which each row is composed of a question's title and body. Additionally, it is necessary to preprocess the text in order to filter out irrelevant information. We start by removing all the code snippets (*i.e.,* enclosed in $< code >$ tag), HTML tags (such as <p> and </p>) and URLs from the corpus. Then, we proceed with the removal of numbers and punctuation marks as they add a little value to the text relevance. We also remove English stop words and add an extra set of stop words composed of the frequently occurring words such as "question", "answer", etc. The full list of removed words is available in our replication package [1]. Finally, we apply lemmatization of words to convert the word to its meaningful base form (*i.e.,* lemma) [3, 23].

To obtain optimal results with LDA, it is necessary to tune its parameters [23, 39]. The first required input for LDA, is the estimated number of *topics* to be generated. Selecting a low value can result in general topics while a high value would produce a long list of detailed topics that could contain noise. Moreover, LDA depends highly on the number of *iterations* that define when the train reaches its end. Hence, we tune the maximum number of iterations through the corpus when inferring the topic distributions. The same uncertainty about the amount of these parameters also exists for the *chunk size* and *passes* as they affect how well/poorly the model can perform. While the chunk size defines the umber of documents to be used at one in each training cycle, the number of passes defines how many times the algorithm is supposed to pass over the whole corpus. Hence, it is important to apply the parameters' tuning [32] for LDA. One the other hand, finding the suitable LDA configuration can be seen as a combinatorial problem where the selection is made from a very large space of choices. Therefore, we use an advanced tuning technique, Genetic Algorithm (GA), to effectively explore this large search space and find the optimal set of parameter values of LDA. GA is a widely used computational search technique, that has proven good performance in solving many software engineering problems [28].

In our adaptation of GA, we selected four parameters to be optimized along with their values ranges, *i.e.,* the number of topics (2 to 50), number of iterations (10 to 5,000), chunk size (10 to 2,000), and passes (1 to 100), . Additionally, we compute and evaluate the LDA models performance using the *Topic Coherence* metric [27] which measures how similar are the topic words are to each other.

Finally, since LDA does provide meaningful names for its generated topics, we manually label the topic names. To implement LDA algorithm, we use the python package `Gensim`[6].

Results. As a result of tuning our LDA model using GA, we found that the (near) optimal configuration consists of fixing the passes, iterations and chunk size to 200, 2,000 and 500, respectively. Additionally, the LDA clustering yields six main topics associated with SO CI questions, that represent the main stages of CI/CD pipeline, namely *Build*, *Testing*, *Version Control*, *Configuration*, *Deployment* and *CI Culture* as represented in Table 4. For each topic, we show a partial set of the associated words (unigrams and bigrams).

Table 4: LDA topics and part of their corresponding words.

Topic	Related Key Words (unigrams & bigrams)
Build	build, job, error, server, branch, fail, jenkins, teamcity, msbuild, time
Testing	test, run, test-case, code, file, xcode, script, integration-test, unit-test, selenium
Deployment	deploy, server, environment, pipeline, jenkins, use, application, web, version, azure-devops
Configuration	configur, file, setup, error, gitlab-ci, install, yml, resource, jenkins, name
Version Control	branch, git, gitlab, github, repository, master, change, push, commit, pull-request
CI Culture	want, need, know, use, way, favorite, important, best, tool company, different

Next, to determine the distribution of each topic, we assign for each question in our dataset, the most dominant such as only topic can be selected.

As shown in Figure 6, we clearly see that, Build topics are the most frequently occurring topics with 39.5% (4,601 questions). In the following, we describe in detail each topic and include some representative examples of associated questions. It should be noted

[6]https://radimrehurek.com/gensim/

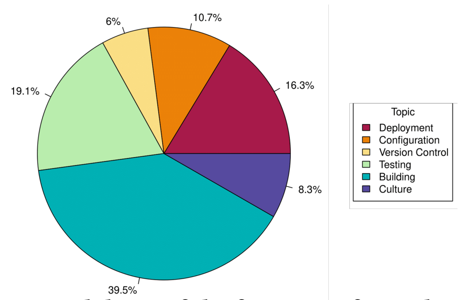

Figure 6: Breakdown of the frequency for each CI topic.

that our analysis of the sub-topics is based on the frequent terms in each topic as well as our manual investigation.

1. Build: As stated by previous research works [17, 28], software build represents a major barrier that developers face when adopting CI. To this aim, developers turn to SO for assistance with build issues. In fact, we encounter in the list of associated words, terms like "issue" and "fail" highlighting that developers seek help mainly with fixing the broken builds (*e.g.,* Quote 1). This suggests that developers struggle to obtain working solutions for build resolution.

How do I fix this incorrect CI build failure?
💬*"".. When the two dependencies change and all projects rebuild we get incorrect failures like the one above. How can we stop from getting these failures?"*

Quote 1: An example of question where the user seeks help with fixing the build failure.[7]

Another common challenge is related to the build performance since the word "time" appeared in the top-10 words as shown in Table 4. As stated in previous works [13, 30], the builds can take hours and even days in CI context. This affects both the speed and cost of software development [21] as well as the productivity of developers who seek help to speed up CI build (cf. Quote 2).

Build in VSO takes a long time
💬*".. However, the execution of the builds takes forever. I have created a small test solution (one class with one property) and a test project (with a single test, using NUnit), and the build takes more than 20 minutes to complete. Is there any way we can speed things up in VSO"*

Quote 2: A Sample question on the need to speed up CI build.[8]

Servers/infrastructures are an essential for adopting CI as they allow the automation of build process. The presence of the terms "jenkins", "teamcity", indicate that these CI servers may not be easily utilized by stakeholders. For instance, the user in Quote 3, is asking how to speed-up the build with Jenkins. This question, despite being viewed 7k times, did not receive a right answer after 8 years of being published.

Jenkins - How to run post-build action without re-running the job?
💬*"I have a lengthy Jenkins job with a failing post-build action. How can I repeatedly run the post-build action without re-running the whole job?"*

Quote 3: Sample question related to Jenkins.[9]

[7]https://stackoverflow.com/questions/7837902
[8]https://stackoverflow.com/questions/29848548
[9]https://stackoverflow.com/questions/15191539

Build systems were developed to automate the code compilation and they are an essential part of CI process. Hence, more effort is needed to improve build systems. In particular, according to Table 4, more attention should be given to MsBuild, the Microsoft build tool, that contains difficult-to-understand features which leads to commonly occurring errors [15]. For instance, in Quote 4, the developer is seeking help to solve an MsBuild error. This question is among the most popular questions in our dataset by reaching 73k views. This finding aligns with Openja et al. [22] results.

build .NET application in Jenkins using MSBuild
💬*".. I've added MSBuild plugin ..But my build processes are failing by showing the below error message"*

Quote 4: Sample question related to MSBuild.[10]

2. Testing: Figure 6 shows that Testing is the second most dominating topic representing 19% of CI questions that developers ask in SO. This topic deals mainly with questions that actively discuss the basics of testing. Indeed, when observing the frequent terms in this topic, we encounter terms related to unit testing, test cases and integration testing. For example, in Quote 5, the user wants to know how to run integration tests. The accepted answer suggested to *"run integration tests automatically on TeamCity agent after main build is completed"*.

How to run integration tests?
💬*"... For plain unit tests, we use TeamCity for continuous integration. How do you run the integration unit tests and when do you run them?"*

Quote 5: Sample question related to integration tests.[11]

Some questions that may be interesting to investigate are questions related to mobile testing. Specifically, we found that *Xcode*, an Integrated Development Environment (IDE) for mac-OS, is a frequent term in testing topic. Indeed, mobile apps undergo frequent updates to introduce new features, fix reported issues or adapt to new technological or environment changes. Hence, introducing changes in this context is risky and can harmfully affect the application rating and competitiveness. Thus, ensuring that the changes can by safely integrated (*i.e.,* by testing them) is of crucial importance. As a result of our analysis, we found that developers search information about how to solve their problems related to testing or setting a mobile test environment as shown in Quote 6.

Travis CI Android Tests: no connected devices
💬*"I am trying to set up Travis for Android. Running the build seems to work so far, but when it comes to the tests, it complains about no connected devices!.."*

Quote 6: Sample question related to mobile testing.[12]

In addition to mobile apps, developers seem to be interested to the continuous testing of Web applications. Specifically, we found that *Selenium*, a set of tools and libraries for supporting Web browser automation, is a frequent term in this topic. In Quote 7, we highlight an example of a question about running Selenium tests with CI:

[10]https://stackoverflow.com/questions/10227967
[11]https://stackoverflow.com/questions/2552506
[12]https://stackoverflow.com/questions/31264136

How to run Selenium tests with CI (Continuous Integration)?
💬*"..I'm using Selenium for automated testing my websites. I have around 100 test cases and I want to run them every day by making Test Suite automatically. "*

Quote 7: A question to test Web applications with Selenium.[13]

3. Deployment: Continuous Deployment (CD) is an essential practice consisting of automatically deploying every change into the production environment [31]. We found that this topic covers 16% of discussions about CI. When analysing the related questions, we found that more than 70% of the discussions on this topic include *"how to"* suggesting that generally, users are seeking help with performing specific tasks about deployment. This result aligns with the findings of Openja et al. [22] who find that the majority of *"how"* questions are related to Deployment topic.

A manual investigation of a significant sample of questions, that represents a confidence level of 99% and an interval of 5%, reveals that users enquire about the deployment pipelines and their associated tools. Indeed, the success of implementing CI/CD practices depends heavily on the appropriate selection of tools/infrastructures [31]. By observing the frequent terms in this topic, we encounter "jenkins" and "azure-devops", two popular CI/CD servers. While we found 357 (or 18% of deployment questions) Jenkins tagged questions, Azure (previously known as Team Foundation Server) tagged questions represent 16.7%. Quotes 8 and 9 show the most popular questions about these two tools (in terms of views count).

JENKINS how to deploy artifacts to maven repo?
💬*"I use jenkins 1.500 and I looking for plugin that will provide possibility to deploy artifacts to maven repository, in previuos version of jenkins it was possible in post build actions using maven-plugin but for now that option dissappear.."*

Quote 8: Sample question about deploying artifacts with Jenkins.[14]

How to set Azure pipeline variable from PowerShell
💬*"..am trying to set the Azure pipeline variable value in PowerShell. I have created one variable winversion in the Azure pipeline. Now, in a PowerShell task, I want to assign some values to the winversion variable. My simple question is how can I change the value of an Azure PipeLine variable at run time?"*

Quote 9: Sample question asking how to setup Azure pipeline.[15]

Web deployment is a relevant sub-topic in Deployment since the terms "web" and "application" appear the frequent LDA terms in this topic. When analyzing the deployment questions, we found that 17% of discussions encounter the term "web". An example of Web deployment question is presented in Quote 10.

Is there any stable tool for complete Web deployment & CI
💬*"I've spent a plenty of hours trying to find a full stable solution for an application deployment (in my case it's php). There are a lot of SO answers, where phing / capistrano / hudson are being proposed, but such propositions make me feel sad.."*

Quote 10: Sample question about Web deployment.[16]

[13]https://stackoverflow.com/questions/5627070
[14]https://stackoverflow.com/questions/14694696
[15]https://stackoverflow.com/questions/55472792
[16]https://stackoverflow.com/questions/35995990

4. Configuration: Configuration management is an essential part of CI/CD pipeline and discussions on this topic cover around 10% of CI questions in our dataset. By observing the frequent terms in this topic, we realize that most of the concerns about configuration is related to the appropriate setup of infrastructures/servers. Indeed, we encounter the terms "jenkins" and "gitlab-ci". By analysing the questions in this topic, we found many questions related to configuration issues with Jenkins. For instance, in Quote 11, the user is asking how to configure Jenkins to run on port 80. This question despite being viewed 77k times, did not receive accepted answers.

How to configure Jenkins to run on port 80
💬*"..Is this because jenkins is running as the jenkins user on a privileged port? If so, how do I fix this? Any other ideas a welcome."*

Quote 11: Sample question about the configuration of Jenkins.[17]

Gitlab CI is a popular service that allows build up CI/CD pipelines. According to our analysis, it seems that configuring the Gitlab CI tool is not trivial. For example, in Quote 12, the developer is facing an issue with the configuration file (*i.e.,* gitlab-ci.yml) of Gitlab CI that does not execute the scripts of build, deployment and testing. In the accepted answer, it was mentioned that the developer should use local paths in order to recognize the directory of the scripts.

gitlab-ci.yml not executing shell script
💬*"I set up gitlab-ci for my project, and inserted the following yml script..Do I have the wrong setup? Obviously its not the syntax and the permissions are allright, otherwise i'd get an error.What could this be?"*

Quote 12: Sample question about the configuration of Gitlab-CI.[18]

5. CI Culture: To take full advantage of CI, a set of guiding principles should be applied. In this topic that represents 8% of discussions in our dataset, developers discuss how to properly embed CI in their companies/projects. For instance, the developer in Quote 13 seeks to gain deeper understanding of the importance of using CI. The accepted answer of this question includes: 💬*"Using CI is a useful skill to have, but you want to avoid developing any bad habits that wouldn't translate to a team environment".*

Is Continuous Integration important for a solo developer?
💬*"h've never used CI tools before, but from what I've read, I'm not sure it would provide any benefit to a solo developer that isn't writing code every day. First - what benefits does CI provide to any project? Second - who should use CI? Does it benefit all developers?"*

Quote 13: Sample question for CI culture topic aims to understand how CI is important.[19]

This topic includes also discussions in which developers establish comparisons about different tools of CI. For example, in Quote 14, the developers need to choose between two popular CI servers, Jenkins and Travis CI. This question has been viewed 132k times.

Jenkins vs Travis-CI. Which one would you use for a Open Source project?
💬*"For my project I need to choose between Jenkins and Travis-CI. I've been using Jenkins for years but I've also read good reviews about Travis-CI. Which one would you use for an Open Source project? What are the main benefits or advantages of both?"*

[17]https://stackoverflow.com/questions/9330367
[18]https://stackoverflow.com/questions/33807387
[19]https://stackoverflow.com/questions/130592

Quote 14: Sample question for CI culture topic aims to compare between Jenkins and Travis CI.[20]

6. Version Control: A core practice of CI is that all developers commit to the mainline (or master) branch daily. This topic is related to challenges in setting up repository branches and maintaining their synchronisation and covers around 6% of questions in our dataset. By observing the topic related words, we found "gitlab" and "github" are among the most used terms which indicates that developers are facing challenges when using these two version control systems. For example, in Quote 15, the user wants to find out a way to force other developer to mention the issue in the commit message on GitHub that seems to be a missed feature in GitHub. Additionally, the question did not receive any accepted answer after 9 years.

How to avoid developers to commit without mention the issue on commit message on Github

".. Our project is currently hosted on GitHub, and we have a well configured Jenkins CI Server too. The doubt is: how we can force our developers to mention a issue before commit?"

Quote 15: Sample question for Verion Control topic.[21]

3.2.3 RQ3. (Challenges) Which topics are the most popular and difficult among CI related questions?

Approach. Similarly to previous studies [4, 22, 23], we consider the (1) view count, (2) favorite count, and (3) score of CI questions as metrics to measure the topic popularity. Of high scores obtained for these metrics, the most popular is the CI topic. In addition, we aim to identify the most difficult/challenging for developers. Particularly, we look at questions that do not have any (accepted) answers as well as the median time the community takes to provide an acceptable answer to a question. Finally, we perform a qualitative analysis by examining a significant sample of unanswered questions (*i.e.,* questions without an accepted and non-accepted answer post). This sample includes 498 questions and represents a confidence level of 99% and an interval of 5% for each topic, from a total of 1,981 unanswered questions.

Results. Table 5 shows the topics popularity and difficulty results.

Popularity. Looking at the table, we clearly see that the *Build* topic is the most popular in terms of favourite, view and score counts while the *Configuration* topic is the least popular. This finding confirms that build issues are the main concerns of CI developers who turn to the SO community to seek help. Next, we compare the popularity of CI questions against the other discussions on SO that are not part of our dataset. As a result, we found that CI questions have a higher average of scores (2.94) compared to the average score of non-CI questions (2.1). However, the views and favorites of non-CI questions are higher by reaching average values of 2472.2 (compared to 1996.5) and 2.7 (compared to 0.89) respectively. This suggests that CI discussions are not among the popular discussions.

Difficulty. When it comes to the questions' difficulty, we found that *Version Control* discussions have the highest rate of unanswered questions among the six topics reaching 21% and 59% of the questions having no answers or any accepted answer respectively. With regards to time needed to receive an accepted answer, we found

Table 5: Popularity and difficulty of CI topics.

Topic	Popularity Metrics			Difficulty Metrics				
	Average Counts			Questions without answers		Questions without accepted answer		Median hours to an accepted answer
	Favorite	View	Score	count	Percentage	count	Percentage	
Culture	0.92	2082.8	3.23	164	17%	536	55%	5.0
Version Control	0.88	2045.9	3.04	148	**21%**	416	**59%**	5.0
Deployment	0.86	1380.5	2.16	349	18%	1102	58%	12.5
Testing	0.94	1642.0	3.00	411	18%	1235	56%	**18.0**
Build	**0.95**	**2488.8**	**3.28**	669	15%	2295	50%	10.0
Configuration	0.61	1654.6	2.45	240	19%	675	54%	10.0
Average for all CI questions	*0.89*	*1996.5*	*2.94*	*1981*	*17%*	*6259*	*54%*	*11*

that *Testing* questions take over 18 hours in median to receive an accepted answer. These results suggest that these two topics are the most challenging for CI developers to answer. At the same time, questions related *CI Culture* seem to be the least challenging by achieving the lowest average percentage of questions without answers (17%) and of median hours to receive an accepted answer (only 5 hours). This may be explained by the fact that usually these types of questions are non-technical (*e.g.,* comparison between two CI servers) and not specific to the developer's project, which make it easier for developers to answer. Overall, we see that CI questions receive accepted answers within a short period of time (11 hours) and that only 17% of the questions remain unanswered. In the following, we investigate some of these unanswered questions.

Unanswered questions. As a final step of our analysis, we examine the unanswered questions (*i.e.,* questions without any accepted or non-accepted answer). To this aim, we manually reviewed a statistically significant sample of unanswered from each topic with a confidence level and interval of 99% and 5% respectively. Specifically, we examined these unanswered questions for incompleteness (*e.g.,* lack of source code, concrete examples, etc.) and ambiguity (unclear or short questions) to identify the reasons behind the poor interaction of users with these questions.

The result of our examination reveals that about 60% of these questions were not completely ignored by users but rather received a least one comment (up to 10 comments per question). We found that these comments are either suggesting some solutions/answers to address the questions (*e.g.,* QuestionID = 45439753[22]) or requesting for more clarification (*e.g.,* in QuestionID = 55527078[23]). In other comments, the users mention that is no solution for the problem (*e.g.,* QuestionID = 8041785[24]). Moreover, since our dataset covers the questions until March 2020, we found some questions being already answered after this date (*e.g.,* QuestionID = 38680366[25]). Finally, we found rare cases where the unanswered questions are not related to CI which means that in these questions, the developers misuses the "continuous integration" tag in the question (*e.g.,* QuestionID = 40231075[26]). Hence, it seems that most of these questions are not actually ignored by the CI community.

4 DISCUSSION AND TAKEAWAYS

More research on build (failure, time and systems). Despite the significant number of studies that investigated the factors behind build failures [7, 21, 26] and the proposed automatic prediction tools [16, 28, 37], CI developers still seek assistance to deal with

[20] https://stackoverflow.com/questions/32422264

[21] https://stackoverflow.com/questions/13704498

[22] https://stackoverflow.com/questions/45439753

[23] https://stackoverflow.com/questions/55527078

[24] https://stackoverflow.com/questions/28741142

[25] https://stackoverflow.com/questions/38680366

[26] https://stackoverflow.com/questions/40231075

build failures (as revealed by our findings); which suggests that more effort is required to investigate/prevent the reasons behind the build breakage. Future research can leverage the history of discussions on SO to examine the possible reasons behind the build failure. Moreover, our findings in RQ2 reveal that the long CI build time is another challenge faced by developers who seek help with solutions to speed up CI process. Recently, some solutions have been proposed to partially address the problem, by detecting the changes that can be skipped during the build [2, 30]. Nevertheless, more effort is needed to speed up the time needed for changes that cannot be skipped. Finally, as revealed in RQ2, the discussions around build systems (*e.g.,* MSbuild) are emergent. This potentially indicates a need for research in building systems as they are an important part of CI practices, as being under-studied [6].

Enhance the user experience. Developers use a variety of tools, including servers, Version Control Systems (VCS), testing frameworks and build systems, to support the CI of their software changes. In this paper, we showed that most of the tags in the CI questions are related to infrastructures and tools. Moreover, we revealed that some tools/infrastructures are emergent in developers' discussions. We, therefore, believe that tool builders must ensure that their products exhibit an optimal user experience. Specifically, the wide adoption of CI depends largely on the high availability of tools/infrastructures. We encourage project stakeholders and developers to pay more attention to the selection of appropriate tools and infrastructures and their configurations including those identified in our study (*e.g.,* Jenkins, TeamCity, Docker, etc.), for building CI pipelines in order to mitigate its challenges and potentially reduce obstacles for developers.

Improve testing activity. Getting people to write tests has been broadly recognized as difficult [36]. This finding is confirmed by our study as developers usually inquire about the basics of testing in the context of CI. Hence, we believe that testing tools builders should summarize more detailed instructions to help developers create their tests more easily and quickly.

Investigate deployment issues. Our analysis revealed that deployment is a popular topic in CI discussions. Similarly to CI, Continuous Deployment (CD) practice helps reduce errors and speed up the development process [31] which motivates the need to study its specific challenges. Having provided a methodology to investigate CI challenges, our study can be extended for an indeeper investigation of deployment challenges faced by developers.

Research on the configuration of CI environments. Our study revealed that the configuration of CI servers/infrastructure is a large concern for developers. While it is been reported that CI systems are vulnerable to misconfiguration [14], little is known how the features are misused in CI specification files. Hence, we encourage researchers to conduct empirical studies on this matter in order to improve the management and verification of CI configurations.

More attention should be paid to Web/mobile platforms. In RQ2, we found Web-related keywords (*e.g.,* "Selenium") in testing and deployment topics (Table 4). Moreover, the emergence of the mobile development is also apparent in these two topics (*e.g.,* "Xcode" and "application" keywords). This suggests a trend toward the CI of software for Web and mobile platforms. At the same time, we note the very limited academic research being done today in supporting the adoption in Web/mobile domains. Hence, we believe that research effort is required to study the specific needs, related to CI adoption, of developers in these platforms.

Lack of expertise and skill. CI/CD pipelines can be difficult for newcomers. The SO community might need to propose incentives to encourage CI experts to further contribute. Our findings motivate more experienced developers to explore the CI-related problems discussed by practitioners and solutions for these problems.

Towards a better understanding of CI practices/principals. 8% of CI questions discuss how to get into the philosophy of CI. This finding may indicate the limited availability of knowledge and training for CI and therefore suggests that it is still challenging for developers to practice CI effectively. Hence, educators are encouraged to ensure that CI principals/practices are covered and practiced in course materials and laboratories.

Teaching different CI topics. We recommend educators to further focus on the most popular and difficult topics based on specific experiences occurred to developers and discussed in Stack Overflow (*e.g.,* build, testing, version control, etc.) as identified in our study and ensure that these issues are covered in course materials and/or practiced by students in labs and/or assignments.

5 THREATS TO VALIDITY

Threats to internal validity are concerned with the factors that could have affected the validity of our results. First, we entirely relied on the existence of the term "continuous integration" in the tag or in the title to identify CI related posts in SO. There is the possibility that we may have missed other CI posts by excluding synonymous terms. However, we highly decreased the false positives (*i.e.,* selected questions that are not really related to CI). Indeed, in rare cases, we found that "continuous integration" term was missused by developers when posting the question. Hence, we believe that our approach resulted in a significantly relevant dataset.

Threats to construct validity are mainly related to the rigor of the study design. The use of LDA to cluster CI discussions can be considered as a threat to the construct validity of our study. However, as mentioned earlier in the paper, this technique has been widely used in similar contexts [22, 23]. Additionally, the search space used to tune LDA parameters could introduce some bias in our results as considering different ranges/parameters may yield to different results. To mitigate this threat, we used an advanced technique, Genetic Algorithm (GA) widely used for the automatic tuning of Machine Learning (ML) techniques [28, 35, 38]. Nevertheless, future replication of this work should explore other ranges/parameters and their impacts on LDA performance. In terms of qualitative analysis, we rely heavily on manual analyzing. Due to the large volume of our data, we selected significant sample of tags (RQ1) and unanswered questions (RQ3) as representative data for our manual analysis with a confidence level and interval of 99% and 5%, respectively.

Threats to external validity concerns the possibility to generalize our results. To conduct our study, we collected data from Stack Overflow (SO), the most popular Q&A forum [22], within a period of 12 years (from 2008 to 2020). However, we cannot guarantee the generalizability of our findings to other forums/websites. Future work should replicate our study considering other technology-based question and answer websites.

6 CONCLUSION AND FUTURE WORK

In this work, we present the first empirical study to investigate the trends, topics and challenges discussed by developers on Stack Overflow (SO) when adopting Continuous Integration (CI). By analysing the asked questions, we show that SO has been widely used by developers to seek assistance with CI challenges. Specifically, Servers and Platforms are tagged the most in CI questions. With regards to CI topics, we found, using tuned LDA modeling, that discussions on CI can be categorized into six topics where 40% of the questions are about "Build" topic. Next, we investigate the characteristics of answers in terms of popularity (*e.g.,* number of views) and difficulty (*e.g.,* hours to receive an accepted answer) and find that "Build" topics are the most popular, while "Version Control" and "Testing" topics seem to be the most difficult. Based on our quantitative and qualitative analysis, we also distill many takeaways for different stakeholders. We plan, as a future work to conduct a survey with different CI stakeholders from both open-source and industry in order to complement our empirical study with further insights into CI adoption and its related challenges, *e.g.,* we can ask the interviewed persons about the topics we observed in our empirical study and how difficult they are for them.

REFERENCES

[1] 2023. Replication package. https://figshare.com/s/9682e9a121a2e51730ad.

[2] Rabe Abdalkareem, Suhaib Mujahid, Emad Shihab, and J. Rilling. 2019. Which Commits Can Be CI Skipped? *IEEE Transactions on Software Engineering* (2019).

[3] Ahmad Abdellatif, Diego Costa, Khaled Badran, Rabe Abdalkareem, and Emad Shihab. 2020. Challenges in chatbot development: A study of stack overflow posts. In *17th International Conference on Mining Software Repositories*. 174–185.

[4] Eman Abdullah AlOmar, Diego Barinas, Jiaqian Liu, Mohamed Wiem Mkaouer, Ali Ouni, and Christian Newman. 2020. An exploratory study on how software reuse is discussed in stack overflow. In *International Conference on Software and Software Reuse*. Springer, 292–303.

[5] Sebastian Baltes, Lorik Dumani, Christoph Treude, and Stephan Diehl. 2018. Sotorrent: Reconstructing and analyzing the evolution of stack overflow posts. In *15th international conference on mining software repositories*. 319–330.

[6] Anton Barua, Stephen W Thomas, and Ahmed E Hassan. 2014. What are developers talking about? an analysis of topics and trends in stack overflow. *Empirical Software Engineering* 19, 3 (2014), 619–654.

[7] Moritz Beller, Georgios Gousios, and Andy Zaidman. 2017. Oops, my tests broke the build: An explorative analysis of Travis CI with GitHub. In *IEEE/ACM International Conference on Mining Software Repositories*. 356–367.

[8] João Helis Bernardo, Daniel Alencar da Costa, and Uirá Kulesza. 2018. Studying the impact of adopting continuous integration on the delivery time of pull requests. In *International Conference on Mining Software Repositories*. 131–141.

[9] David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent dirichlet allocation. *the Journal of machine Learning research* 3 (2003), 993–1022.

[10] Adam Debbiche, Mikael Dienér, and Richard Berntsson Svensson. 2014. Challenges when adopting continuous integration: A case study. In *International Conference on Product-Focused Software Process Improvement*. Springer, 17–32.

[11] Omar Elazhary, Colin Werner, Ze Shi Li, Derek Lowlind, Neil A Ernst, and Margaret-Anne Storey. 2021. Uncovering the benefits and challenges of continuous integration practices. *IEEE Transactions on Software Engineering* (2021).

[12] Martin Fowler. 2006. Continuous Integration. https://www.martinfowler.com/articles/continuousIntegration.html,. Accessed: 2020-01-01.

[13] Taher Ahmed Ghaleb, Daniel Alencar da Costa, and Ying Zou. 2019. An empirical study of the long duration of continuous integration builds. *Empirical Software Engineering* (2019), 1–38.

[14] Volker Gruhn, Christoph Hannebauer, and Christian John. 2013. Security of public continuous integration services. In *9th International Symposium on open collaboration*. 1–10.

[15] Sayed Y Hashimi and S. I. Hashimi. 2006. The Unified Build Engine: MSBuild. *Deploying. NET Applications: Learning MSBuild and ClickOnce* (2006), 21–43.

[16] Foyzul Hassan and Xiaoyin Wang. 2017. Change-aware build prediction model for stall avoidance in continuous integration. In *ACM/IEEE International Symposium on Empirical Software Engineering and Measurement*. 157–162.

[17] Michael Hilton, Nicholas Nelson, Timothy Tunnell, Darko Marinov, and Danny Dig. 2017. Trade-offs in continuous integration: assurance, security, and flexibility. In *11th Joint Meeting on Foundations of Software Engineering*. 197–207.

[18] Michael Hilton, Timothy Tunnell, Kai Huang, Darko Marinov, and Danny Dig. 2016. Usage, Costs, and Benefits of Continuous Integration in Open-source Projects. In *Int. Conference on Automated Software Engineering*. 426–437.

[19] Eero Laukkanen, Juha Itkonen, and Casper Lassenius. 2017. Problems, causes and solutions when adopting continuous delivery—A systematic literature review. *Information and Software Technology* 82 (2017), 55–79.

[20] Eero Laukkanen, Maria Paasivaara, and Teemu Arvonen. 2015. Stakeholder perceptions of the adoption of continuous integration–a case study. In *2015 agile conference*. IEEE, 11–20.

[21] Yang Luo, Yangyang Zhao, Wanwangying Ma, and Lin Chen. 2017. What are the Factors Impacting Build Breakage?. In *14th Web Information Systems and Applications Conference (WISA)*. 139–142.

[22] Moses Openja, Bram Adams, and Foutse Khomh. 2020. Analysis of Modern Release Engineering Topics:–A Large-Scale Study using StackOverflow–. In *International Conference on Software Maintenance and Evolution*. 104–114.

[23] Anthony Peruma, Steven Simmons, Eman Abdullah AlOmar, Christian D Newman, Mohamed Wiem Mkaouer, and Ali Ouni. 2022. How do i refactor this? An empirical study on refactoring trends and topics in Stack Overflow. *Empirical Software Engineering* 27, 1 (2022), 1–43.

[24] Gustavo Pinto, Fernando Castor, Rodrigo Bonifacio, and Marcel Rebouças. 2018. Work practices and challenges in continuous integration: A survey with Travis CI users. *Software: Practice and Experience* 48, 12 (2018), 2223–2236.

[25] Akond Rahman, Amritanshu Agrawal, Rahul Krishna, and Alexander Sobran. 2018. Characterizing the influence of continuous integration: Empirical results from 250+ open source and proprietary projects. In *4th ACM SIGSOFT International Workshop on Software Analytics*. 8–14.

[26] Thomas Rausch, Waldemar Hummer, Philipp Leitner, and Stefan Schulte. 2017. An empirical analysis of build failures in the continuous integration workflows of Java-based open-source software. In *MSR*. 345–355.

[27] Michael Röder, Andreas Both, and Alexander Hinneburg. 2015. Exploring the space of topic coherence measures. In *ACM international conference on Web search and data mining*. 399–408.

[28] Islem Saidani, Ali Ouni, and Mohamed Wiem Mkaouer. 2022. Improving the prediction of continuous integration build failures using deep learning. *Automated Software Engineering* 29, 1 (2022), 1–61.

[29] Islem Saidani, Ali Ouni, Mohamed Wiem Mkaouer, and Fabio Palomba. 2021. On the impact of Continuous Integration on refactoring practice: An exploratory study on TravisTorrent. *Information and Software Technology* 138 (2021), 106618.

[30] Islem Saidani, Ali Ouni, and Wiem Mkaouer. 2021. Detecting skipped commits in continuous integration using multi-objective evolutionary search. *IEEE Transactions on Software Engineering* (2021).

[31] Mojtaba Shahin, Muhammad Ali Babar, and Liming Zhu. 2017. Continuous integration, delivery and deployment: a systematic review on approaches, tools, challenges and practices. *IEEE Access* 5 (2017), 3909–3943.

[32] Chakkrit Tantithamthavorn, Shane McIntosh, Ahmed Hassan, and Kenichi Matsumoto. 2018. The impact of automated parameter optimization on defect prediction models. *IEEE Transactions on Software Engineering* 45, 7 (2018), 683–711.

[33] Bogdan Vasilescu, Yue Yu, Huaimin Wang, P. Devanbu, and Vladimir Filkov. 2015. Quality and Productivity Outcomes Relating to Continuous Integration in GitHub. In *Joint Meeting on Foundations of Software Engineering*. 805–816.

[34] Carmine Vassallo, Sebastian Proksch, Harald C Gall, and Massimiliano Di Penta. 2019. Automated reporting of anti-patterns and decay in continuous integration. In *41st International Conference on Software Engineering*. 105–115.

[35] Ananto Setyo Wicaksono and Ahmad Afif Supianto. 2018. Hyper parameter optimization using genetic algorithm on machine learning methods for online news popularity prediction. *Int. J. Adv. Comput. Sci. Appl* 9, 12 (2018), 263–267.

[36] David Gray Widder, Michael Hilton, Christian Kästner, and Bogdan Vasilescu. 2019. A conceptual replication of continuous integration pain points in the context of Travis CI. In *27th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*. 647–658.

[37] Zheng Xie and Ming Li. 2018. Cutting the Software Building Efforts in Continuous Integration by Semi-Supervised Online AUC Optimization.. In *IJCAI*. 2875–2881.

[38] Li Yang and Abdallah Shami. 2020. On hyperparameter optimization of machine learning algorithms: Theory and practice. *Neurocomputing* 415 (2020), 295–316.

[39] Xin-Li Yang, David Lo, Xin Xia, Zhi-Yuan Wan, and Jian-Ling Sun. 2016. What security questions do developers ask? a large-scale study of stack overflow posts. *Journal of Computer Science and Technology* 31, 5 (2016), 910–924.

[40] Mansooreh Zahedi, Roshan Namal Rajapakse, and Muhammad Ali Babar. 2020. Mining questions asked about continuous software engineering: A case study of stack overflow. In *Evaluation and assessment in software engineering*. 41–50.

[41] Fiorella Zampetti, Carmine Vassallo, Sebastiano Panichella, Gerardo Canfora, Harald Gall, and Massimiliano Di Penta. 2020. An empirical characterization of bad practices in continuous integration. *Empirical Software Engineering* 25, 2 (2020), 1095–1135.

[42] Yangyang Zhao, Alexander Serebrenik, Yuming Zhou, Vladimir Filkov, and Bogdan Vasilescu. 2017. The Impact of Continuous Integration on Other Software Development Practices: A Large-scale Empirical Study. In *IEEE/ACM International Conference on Automated Software Engineering*. 60–71.