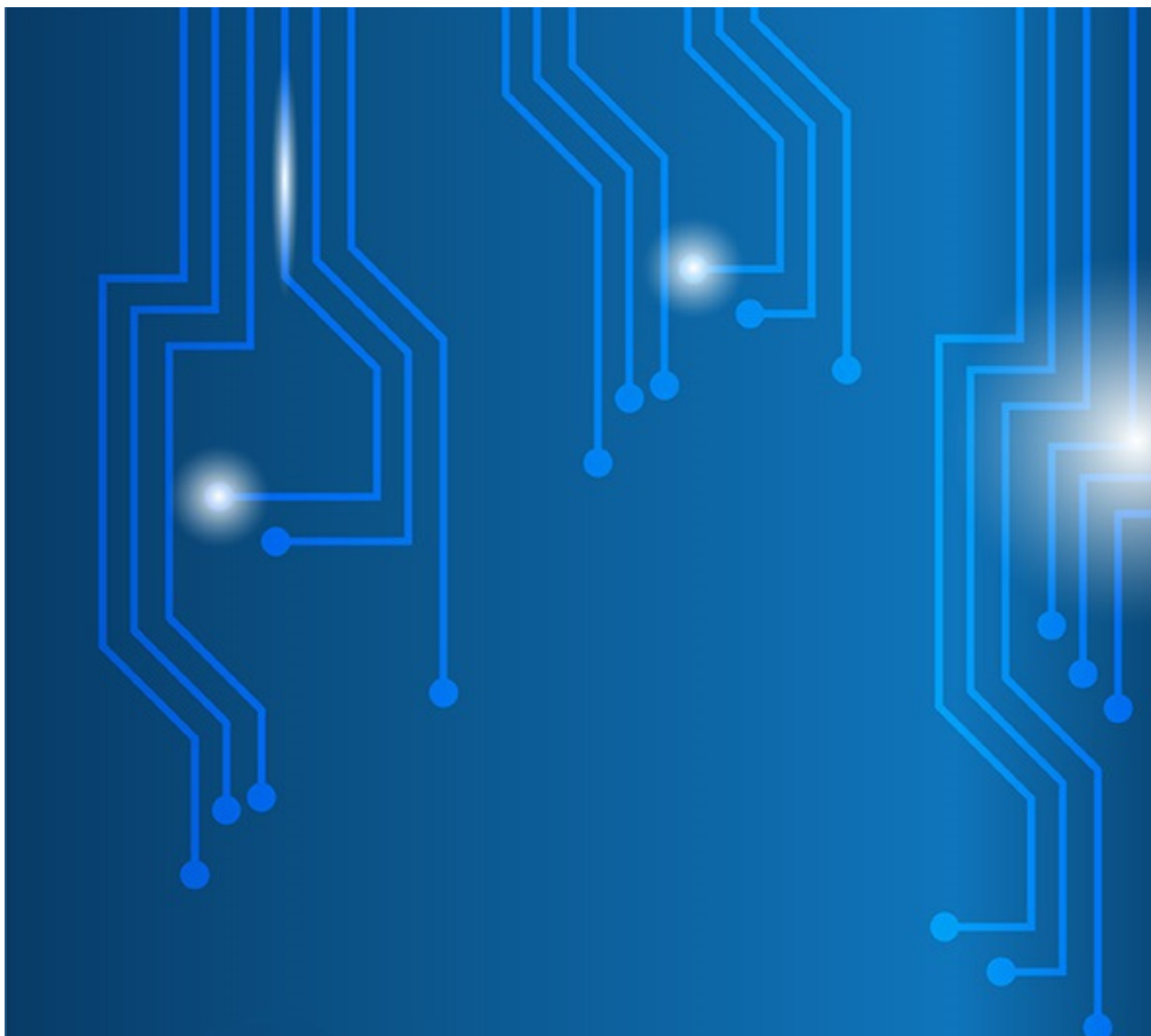




**Journal of University of Information Science
and Technology "St. Paul the Apostle" - Ohrid
ISSN: 2955-182X**

Publisher University of Information Science and Technology "St. Paul the Apostle" - Ohrid

Volume 5 | Issue 1 | 2025

EDITORIAL - Preface to Volume 5 Issue 1 of the Journal of University of Information Science and Technology “St. Paul the Apostle” – Ohrid

Dear Readers,

Journal of University of Information Science and Technology “St. Paul the Apostle” – Ohrid is an international, peer-reviewed journal published annually since 2016. It is an open access Journal available at the website of the University of Information Science and Technology “St. Paul the Apostle” – Ohrid (<https://journal.uist.edu.mk/>)

This Journal is committed to publishing and disseminating high-quality and novel scientific research work in the broad field of Applied Mathematics, Automation & Robotics, Blockchain, Business, Economy, Entrepreneurship & Management, Communication & Network Technology, Cybersecurity, Data Science, Distributed and Parallel Computing, Digital Art and New Media Art, Metaverse, Virtual and Augmented Reality, Communication & Network Technology, Smart Systems & IoT, Information Society & Digital Transformation, Biomedical Engineering.

As editor-in-chief of the Journal of the University of Information Science and Technology “St. Paul the Apostle” – Ohrid, I am pleased to present Volume 5, an issue mainly devoted to the decentralized ecosystems, providing insights into blockchain technology, smart contracts, non-fungible tokens and online social networks. We thank all the authors for contributing to this Issue and all the reviewers for providing detailed and timely evaluations of the submitted manuscripts.

It is an excellent challenge for the editor members to bring a new journal issue into the world, primarily when the Journal aims to publish high-quality manuscripts. I want to express my sincere gratitude to all of them for their excellent work, remarkable contribution, enthusiasm and support.

Sincerely Yours,

Editorial Board

February 2025



Journal of University of Information Science
and Technology "St. Paul the Apostle" - Ohrid
ISSN: 2955-182X

IMPRESUM

FOUNDER AND PUBLISHER

University of Information Science and Technology "St. Paul the Apostle" – Ohrid

EDITOR IN CHIEF

Professor Ninoslav Marina, PhD

EXECUTIVE EDITOR

Associate Professor, Atanas Hristov, PhD

EDITORIAL ADVISORY BOARD

Professor Ishwar K. Sethi, PhD Oakland University, USA

Professor PREM K. KALRA, PhD Head & Professor, Indian Institute of Technology, Kanpur, India

Professor Eric Châtelet, PhD Université de technologie de Troyes, France

Professor Weiler A. Finamore, PhD Pontifical Catholic University of Rio de Janeiro, Brazil

Professor Katsumi Yamashita, PhD University of Osaka, Japan

Professor Mohammad Reza Asharif, PhD University of the Ryukyus, Okinawa, Japan

Professor Heung Gyoon Ryu, PhD Chungbuk National University, Seoul, South Korea

Professor Ivan Bimbilovski, PhD UIST "St. Paul the Apostle" – Ohrid, Macedonia

Abdel-Badeeh M. Salem, PhD Professor of Computer Science, Faculty of Computer and Information Sciences, Ain Shams University, Cairo, Egypt

EDITORIAL BOARD

Associate Professor, Dijana Capeska Bogatinoska, PhD

Associate Professor, Aneta Velkoska, PhD

Associate Professor, Jovanka Damoska Sekuloska, PhD

Associate Professor, Anis Sefidanovski, PhD

Associate Professor, Jane Bakreski, PhD

Assistant Professor, Goran Shibakovski, PhD

Assistant Professor, Aleksandar Karadimche, PhD

Assistant Professor, Mersiha Ismajloska, PhD

Assistant Professor, Ile Dimitrieski, PhD

Assistant Professor, Rasim Salkoski, PhD

Assistant Professor, Jovan Karamachoski, PhD

Assistant Professor, Ljubinka Sandjakoska, PhD

Assistant Professor, Daniela Mechkaroska, PhD

ISSN 2955-182X

CONTENTS

M. Grabuloski, R. Salkoski, E. Bronja, A. Salkoska

Efficient automated software testing strategies and methods 5

A. Kotevski

Spam detection pipeline using AI and NLP 16

I. Boseski, I. Petrov, K. Jakimoski, S. Kitanov, A. Ilieva

Evolution of Channel Parameters' Performance in an Urban Setting for Next Generation
Mobile Networks 27

S. Baloska, D. Mechkaroska, E. Domazet

AI-driven Chatbot in E-Commerce 35

Efficient automated software testing strategies and methods

Marko Grabuloski [†]. Rasim Salkoski [‡]. Ensar Bronja [†]. Anita Salkoska [†]

Email(s): marko.grabuloski@cns.uist.edu.mk, rasim.salkoski@uist.edu.mk, ensar.bronja@cns.uist.edu.mk, anita.salkoska@cns.uist.edu.mk

[†] Department of Communication Networks and Security, University of Information Science and Technology, “St. Paul the Apostle”, Ohrid, 6000, Republic of North Macedonia

[‡] Department of Information Systems, Visualization, Multimedia and Animation, University of Information Science and Technology, “St. Paul the Apostle”, Ohrid, 6000, Republic of North Macedonia

Abstract

This research aims to formalize and systematize automated software testing strategies, drawing conclusions on the most effective approaches. It covers key aspects of testing, including its role in the continuous integration and continuous delivery (CI/CD) lifecycle, strategies for selecting and utilizing testing tools, and best practices for writing tests and choosing test scenarios. Special attention is given to Test-Driven Development (TDD), focusing on its flexible use as a tool to accelerate the software development process. Additionally, methods for selecting test scenarios are described, suggesting that engineers should first cover the most common scenarios and then consider edge cases. The formal modeling of software component relationships is also explored to identify the most efficient testing strategies. The goal is to categorize and formalize all facets of automated software testing, providing guidelines for optimal methodologies and suggesting approaches to improve overall software quality.

Keywords

Test-Driven Development, testing, code styles, automated tests.

1 Introduction

Testing is a core part of developing high-quality software. Over time, different types of tests and techniques have been developed for the purpose of testing. These techniques target different parts of the code and are applicable at different stages of the software development lifecycle. Besides asserting the correctness of the code, tests help in maintaining code quality, can increase the speed of development, and serve as documentation for the business requirements and technical decisions. There are several ways to categorize tests. Based on who is executing the tests, there are two types. Manual tests, performed by skilled software testers who identify unseen problems and assist the development process. Automated tests, performed by a machine.

2 Software development and release process

The modern software development process, with time, becomes more and more standardized. This process is supported by different software systems like code repositories, artifact repositories, and continuous deployment systems. During this process, the code and its derivatives are moved around these software systems.

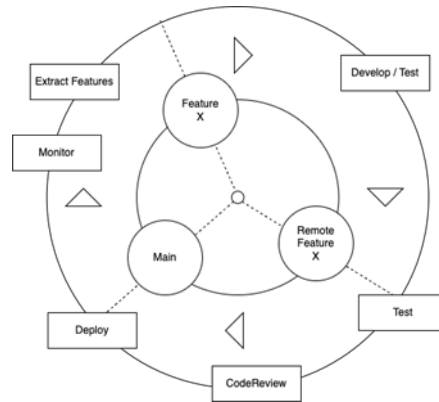


Figure 1: Modern software development release process

The process, called pipeline described in [11], starts with developing new feature in a feature branch, branched from the code's repository main branch. Ideally, the code is developed in parallel while writing tests. This code is tested with automation tests, reviewed by other engagers, tested manually and released. The released feature is monitored for bugs/errors, security problems, and areas of business interest. Depending on the monitoring results and business requirements, new features are created that attend to these areas of interest.

3 Testing practices

A. Selecting testing tools

Before writing any tests, the used styles and tools need to be agreed upon. Before adding, new tools thorough research should be done in their reliability, security, popularity. Popular and trusted components and frameworks are maintained in specialized repositories where bugs and security issues are regularly reported and fixed. New tools techniques should not be used without the team consent, since this writing and maintaining tests becomes harder and inconsistent.

B. Testing code styles

Tests are functions that call parts of the code of the software of interest. The names of the tests should clearly reflect the case that they perform. We prefer is of the following format **when<>And<>...<>Then**. In the placeholders next to **when**, **and**, the conditions under which the test case occurs are placed and the placeholder, next to **then**, the result is written. Other types of naming tests might be **given<>And<>Then**, or **testThat<>When<>And**, same as the **when<>And<>Then**, could be more readable in some cases. Engineers use tests to understand how software works, by reading test code they gain inside into a specific function or overall software functionality. Every test should also be self-sufficient, not depend on others. This makes the test's covered scenarios more understandable. In case of a code change, only a smaller number of tests will be affected. As conclusion:

1. The test name should be descriptive and can be long; it should reflect what the function does.
2. The test code needs to be written with a goal that clearly describes what is tested.
3. Names of variables and helper functions in the tests need to be descriptive and can be long.
4. Tests should contain comments additionally describing what is tested.
5. Extracting common functions and the DRY principle should not be used in every case.

As described in [1], [2] and [10], "Don't repeat yourself" (DRY), is a principle of software development aimed at reducing repetition of information, which is likely to change, replacing it with abstractions that are less likely to change, or using data normalization which avoids redundancy in the first place.

C. Testing phases and comments

Tests contain 3 phases:

1. Preparation phase is when variables are defined, and values are assigned.
 2. Execution phase is when the function that needs to be tested is executed.
 3. Assertion phase is when output of the function is asserted for correctness.
- These phases should be clearly separated, with comments or new lines.

```
// Test 1
public void whenSalaryForMonthReceivesNumberOfDaysThenMultipliesByFixComponent50() {
    // Preparation phase
    int days = 22;
    int fixedComponent = 50;
    // Execution phase
    // calculate the salary for 22 days
    int monthlySalary = salaryForMonth(22);
    // Assertion phase
    // assert the salary for 22 days
    assert fixedComponent == monthlySalary;
}
```

In the above code example, comments with the phase names are added for descriptive and explanatory reasons. These kinds of comments are usually not added in production code.

D. Test coverage

The percentage of unique lines of code from the application under test that are executed during testing is called test coverage. This number is used by teams to quantitatively express and share the code quality. The required coverage can be defined by several entities, for example:

- Companies.
- Teams.
- Stakeholders, like clients and management, based on the software requirements.

The general rule for testing is that the code should be tested close to 100%, meaning every line of code should be tested. Additionally, it is important that there is minimal duplication of tests on the same line.

This number changes during the life cycle of the software. While there are some engineers and teams that really try to have 100% coverage, it's often the business's strict deadlines that push the coverage down.

E. What to test?

It is not required to test 100% of the lines. Instead, while developing the software, it is a good practice to consider all possible scenarios that the software might be used in. To achieve this, while developing the software or writing the tests, the function that is under development needs to be tested in the following scenarios:

1. Mode tests.
2. Domain tests, verify the input belongs to the domain.
3. Edge case tests changing case tests.

F. Testing the Mode

Test the representative examples, the most common, and most relevant for the business and happen most often. This kind of tests make sure that the code works correctly most of the time, and that the most valuable features are tested. Formally, mode describes the most common event that happens in a system or, the most numerous value that a random variable function produces. This is not necessary only one scenario, but the most common ones.

Example 1: Let's assume that We need to develop a function to calculate the monthly salary of an employee based on how many days he/she has worked, assuming that every employ in the company is paid equally.

Let's first consider the scenario, testing component denoted as X . Formally $X(x_{in}) = x_{out}$ would mean that for input x_{in} component X give the result x_{out} . The input of the functions under test can be one or more parameters; they can be strings, integers, or different primitive types or classes made of primitive types or other classes. In general, the parameters are vector-like structures, usually of high dimension. Because of this, there are just too many cases to test; there is no possible way to test all scenarios. To efficiently test there are several strategies that can be applied to narrow down the number of tests and test scenarios. Going back to Example 1, in a single month, some employees might have worked more and some fewer days. It is not recommended to test all the inputs but to pick the most important scenarios. In the example, usually, an employee would work 22 days per month. This is different from the average, where the average will likely be less than 22. Initially, testing the most common scenario, covers most of the components use cases. The mode test can be seen in Test 1.

G. Validating and defining domains

This test scenario would verify the input data, ensuring it makes sense or is in the domain of values. Here, two scenarios can be tested:

1. Verify the input is of the correct data type and data format. These tests are more common in dynamically typed languages and might need additional assertions to verify that the input data is of the correct type. This would ensure that the function receives numbers and not strings, or integers and not real numbers. These kinds of errors are usually reported by the system without any additional checking and need to be handled globally, reporting that the system is misused.
2. Verify that the input is in the domain. Here, although the input datatype is correct, it still does not make sense for the function. In Example 1, this kind of data would be 33 or -10, as it does not make sense that someone has worked 33 days per month or -10 days per month. In the case of this kind of input, the function or component should report the wrong input. In Java, this is done by throwing an exception.

```
// Test 2
@Test
public void
whenSalaryForMonthReceivesSmallerReceivesNegativeInputOfDaysThenThrowsIllegalArgumentEx
ception() {
    // Check lower bound
    assertThrows(IllegalArgumentException.class, () -> {salaryForMonth(-1)});
}
// Test 3
@Test
public void
whenSalaryForMonthReceivesBiggerInputOfDaysThen31ThrowsIllegalArgumentException() {
    // Check upper bound
    assertThrows(IllegalArgumentException.class, () -> {salaryForMonth(32)});
}
```

Additional tests ideally should be written for 31 and 1, which are the boundaries of the domain.

H. Edge cases and challenging cases

So far, we have covered the mode and the domain. The next step would be to consider edge cases and challenging cases. In Example 1, We would consider edge cases 31 and 1. Testing for challenging cases is the most difficult as they can't be categorized; they are exceptions to the rules and difficult to comprehend. These cases can make coding a simple function challenging. What would be a challenging case in this scenario? One problem that might go unnoticed is that the domain is dynamic and depends on the month; different months have a different number of days. Perhaps the function needs to know the month that it calculates the salary for. To explain this more simply, a person might work 31 days in July, but he/she can't work more than 30 days in June. Although, working for 31 days in a month is most probably illegal in every country, it is just physically impossible to work 31 days in June as June has 30 days. On top of this, February has 28 days except if it is a leap year.

The quality of the software depends mostly on whether the engineers have considered the edge cases and the challenging cases.

I. Grouping tests

Having a lot of tests might become difficult to manage. Over time, some tests might become flaky, or they might be written as flaky from the start. One approach is to use parameterized tests, and the other is to have multiple assertions in one test. Parameterized tests run the same test with different parameters, covering almost similar scenarios. They are ideal when more elements from the domain need to be tested under the same scenario. On the downside, they make the testing code more general. When having multiple assertions in one test, tests are more understandable but can become long.

According to [3], A flaky test refers to testing that generates inconsistent results, failing or passing unpredictably, without any modifications to the code under testing.

The previous Test 1 and Test 2 have the same domain and can be merged in one Test 3 using the multiple assertions strategy. Test 3 will look like this:

```
@Test
void
whenSalaryForMonthReceivesValuesNotBetween0And32ThenItThrowsIllegalArgumentException(){
    // Check upper bound
    int workingDaysAboveUpperBound = 32;
    assertThrows(IllegalArgumentException.class, () -> {
        salaryForMonth(workingDaysAboveUpperBound);
    });
    // Check Lower bound
    int workingDaysBelowLowerBound = -1;
    assertThrows(IllegalArgumentException.class, () -> {
        salaryForMonth(workingDaysBelowLowerBound);
    });
    // Check value in bound
    int workingDaysInBound = 20;
    int expectedSalary = 1000;
    int result = salaryForMonth(workingDaysInBound);
    assertEquals(expectedSalary, result);
}
```

Conclusions:

- Tests should be easy to maintain and fast to run.
- Similar test scenarios might be grouped into one using multiple assertions.

4 Test driven development (TDD)

As described in [4], [7], [8] Test-driven development (TDD) is a way of writing code that involves writing an automated unit-level test case that fails, then writing just enough code to make the test pass, then refactoring both the test code and the production code, then repeating with another new test case.

Given Example 1, TDD can be put into practice in the following steps:

```
// 1. Write an empty function:
public void Double salaryForMonth(int days){
    return null
}

// 3. Write test
public void whenSalaryForMonthReceivesNumberOfDaysThenMultipliesByFixComponent50(){
    days = 22
    fixedComponent = 50
    monthlySalary = salaryForMonth(22);
    assert fixedComponent == monthlySalary
}

// 3. Run test, Test will fail as the implementation of salaryForMonth is not correct.
// 4. Refactor/fix salaryForMonth, so the test passes
public void Double salaryForMonth(int days){
    return 50*days
}

// 5. Write additional test
@Test
public void
whenSalaryForMonthReceivesSmallerReceivesNegativeInputOfDaysThenThrowsIllegalArgumentE
xception() {
    // Check Lower bound
    assertThrows(IllegalArgumentException.class, () -> {
        salaryForMonth(-1);
    });
}
```

```

    });
}
// 6. Run test, test will fail since -1 is not allowed as an argument.
// 7. Refactor/fix salaryForMonth, so that the test will pass.

public void Double salaryForMonth(int days){
    if (days < 0) {
        throw new IllegalArgumentException("Input, number of days, must not be smaller than
0");
    }
    return 50*days
}

```

This concept can continue to a Step 8 and 9 will include refactoring the tests by combining the test cases into one single test and refactor the function under test so that the test changes will pass.

A. Conclusion on TDD

The most common practice is to write the code first and then test it. Most of the community, although never using strict TDD, will disagree with this approach as it is considered that “superstar developers” use TDD and “bad, low-quality developers” don’t. Stand on this issue is the following:

1. All test scenarios, required by the business need to be tested.
2. Tests should be used to help in development.
3. Aim for near 100% coverage.

Instead of manually testing code, writing tests while coding will save time. This is because, if tests are not available, engineers would need to manually test the code. This involves steps like starting the entire application, manually navigating the system to execute the preparation phase, and manually asserting. Another scenario would be to debug the entire application, and both are not ideal, time-consuming, and complicated. To respect the rules mentioned before, a good approach would be initially to write part of the code first and then write tests for it. We use the test to make sure that the code does what We want, and We continue coding. We don’t have a strict rule when to write the test and when to write the code. Furthermore, we almost never pile up a lot of code that is not covered with tests or write tests for every new small change in a component. TDD done by the book also reduces the speed of coding because of the constant refactoring of code and tests. A balanced and individually chosen approach that will speed up development and introduce quality is what We advocate for.

5 Which code should be tested?

Let’s formally define several concepts for a component/function under test: If X is the component under test, X_{in} is the set of all written tests for X , and X_{out} is the set of all outputs for X_{in} . This is given by the following relation: $X: X_{in} \rightarrow X_{out}$. A software component is covered by tests if it satisfies the defined criteria from the agreed test coverage; no additional tests are needed.

Let’s formally define a function $coverage(X)$, where X is a function or a software component and $coverage(X)$ are all the tests written for the component X to be covered. This will allow us to more formally try to assert what needs to be tested in the following text.

Conclusion: The component X is covered with the set of tests X_{in} if and only if $coverage(X) \subset X_{in}$

The goal is to test X with the smallest possible set X_{in} that covers X .

The question “Which code should be tested?” might look straightforward, but often unnecessary tests are written. We try to respect these rules:

1. Don’t test external functions or components.
2. Avoid testing the same lines of code using the same type of tests multiple times.
3. Test smaller independent components more extensively and then dependent higher-level components.
4. Tests on dependent components and business cases should test the happy path and expected failures.

First, let’s define what is an internal and external component.

A. Internal and External components

- Internal component: An internal component is any class, function, or other software building block that is developed specifically for the software that is in development.
- External component: Any component or function that is built into the language, framework, or used library is considered an external component.

Examples of external components:

- Arithmetic operations (+, -, *, etc.) and similar functions that are built into the language should not be tested.
- All functions and classes that belong to the language should not be tested. For example, all functions in every Java class and interface, like String, Collection, ArrayList, and functions like System.out.
- All functions and classes that belong to external libraries and frameworks.

Now it is clear that external components are most probably developed by other teams and in other projects, and they should be maintained by these teams.

Conclusions: External components should not be tested. External components should not be added to the software without extensive consideration and consensus from the software development team members.

B. Dependent and Independent components

Let's look at dependent components. Consider a software system made of a component A and components D_i .

- **Independent component:** Let's say A has a defined (required) input a_{in_i} and output a_{out_i} . If all the logic that is required to produce a_{out_i} from any a_{in_i} is inside A , then A is an independent component.
- **Dependent component:** Let's say A has a defined (required) input a_{in_i} and output a_{out_i} . If there is at least one a_{in_i} that produces a different a_{out_i} or is undefined in case one D_i is removed from the software or changed in any way, then A is a dependent component on D_i , and D_i is a dependency of A .

If A is dependent on D , and D is removed from the software, component A will stop functioning correctly.

The dependent component can be dependent on **internal**, **external**, or both internal and external components. Let's consider the relations between components in several scenarios. For the following scenarios let's assume, if not specified else, that all the components X , are covered with test by the set $X_{in} = coverage(X)$.

Scenario 1: Testing single independent internal component.

This scenario is like Test 1. There is only one internal component that is under tests.

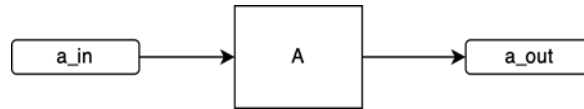


Figure 2: Scenario 1

Scenario 2: Let's consider a scenario where D is an internal component, and D is an exclusive dependency of A , meaning there is no other component that is dependent on D .

The test cases for component A is the set A_{in} , the test cases and test results for component D are D_{in} and D_{out} . The number of test case scenarios for the entire system will be $|D_{in}| + |A_{in}|$. This is not the ideal number of tests, and these are the two reasons why:

1. D is partially tested with A_{in}
2. Tests in D_{in} are not required in this type of system.

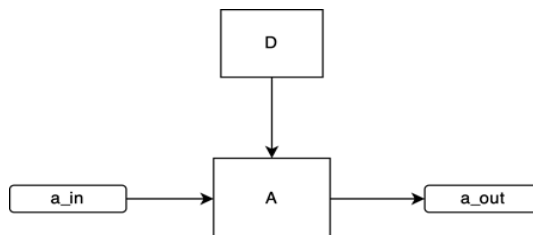


Figure 3: Component A with exclusive dependency D

Let's demonstrate why point 1 is true. First, let's define a set T_d as the set of all possible tests for the D . In this context, D_{in} is a subset of T_d . D is partially tested while running the tests in A_{in} , because D is executed on a subset of T_d . Since D is a dependency of A , there is a subset AD_{in} , $AD_{in} \subset A_{in}$, that is influenced by D . This means that changes in D will change the AD_{out} for AD_{in} . Moving forward, because D influences the tests in the set AD_{in} , there must be an image $DA_{in} \subset T_d$ of AD_{in} since AD_{in} is not necessarily in the domain of D . Since $DA_{in} \subset T$, D is partially tested on A_{in} .

Now let's demonstrate why point 2 is correct. Since D is an exclusive dependency of A , let's assume that point 2 is false. That would mean that D is covered by D_{in} and A is covered by A_{in} and D , and yes, the system is fully covered. But this kind of system can be optimized. Since D is an internal component and a dependency of A , A can be written as a function of D and its input a_{in} . $A(a_{in}, D) = a_{out}$, since D is always a fixed component in the input of A , A can be represented as an independent component A' if D is copied inside A . Now A can be written as $A'(a_{in}) = A(a_{in}, D) = a_{out}$, and D can be completely removed from the system. A' will be completely covered by the tests in A_{in} , and D in this kind of system will not exist. If a component does not exist, no tests are required, meaning D_{in} is not required in this new system, that bring us to a contradiction with the initial assumption.

Conclusions:

1. If A is dependent on D in a system where A and D are covered by sets A_{in} and D_{in} , then D is partially covered by the tests in A_{in} .
2. If A is dependent on D in a system where A and D are covered by sets A_{in} and D_{in} , then a system can be optimized by merging A and D in a single component A' that is fully covered by tests in A_{in} .
3. By merging component with exclusive dependency, the size of tests is equal or reduced.

This would mean that the component D should not be tested in isolation and the code should only rely on the tests for A' .

Components with exclusive dependencies can be merged.

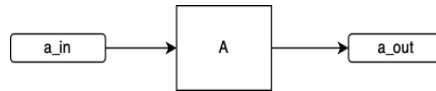


Figure 4: Scenario 2, updated

This is not a hard rule, a common exception can be components that belong to different layered architecture, for example components from a different layer of a MVC architecture should not be merged. Components that belong to the same layer can be merged if they have exclusive dependency. To reduce the number of tests in the scenarios where dependent components are not merged a strategy of mocking is used, described in the scenario.

Scenario 3: D is an external component and A is dependent on D .

In this scenario D is not tested, only A is tested, this is because D is external component. Regardless D is again tested partially by A_{in} . To reduce tests number, different strategy can be used, called Mocking.

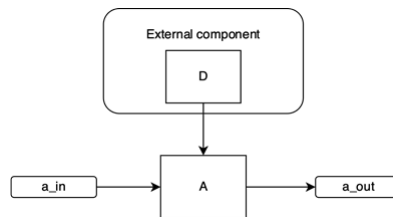


Figure 5: Scenario 3, D is an external component, and A is dependent on D .

Mocking is a strategy where, instead of the component D , a new component D_{mock} is used as a dependency for A .

The D_{mock} has the same function signatures as D . While writing tests, the functions and/or properties of the D_{mock} are overwritten or mocked to return what is needed for the current test scenario.

Example 2: Let's consider a function that calculates the salary for an employee based on the days of the month, given that all employees have different daily fees but are paid for the entire month only based on how many days the month has.

Now let's say that there is an external component, class `Month`, that holds information for a month and is constructed from the index of the month, 1,2,3...12 and has a function `getNumberOfDays` that returns the number of days for this month.

```
public class SalaryCalculation{
    public int getSalaryForMonth(int dailyFee, Month month){
        return dailyFee * month.getNumberOfDays()
    }
}
```

The tests need to verify the business case where `month.getNumberOfDays()` is multiplied with `dailyFee` to get the salary.

```
@Test
public void whenNumberOfDaysIs30AndDailyFeeIs100ThenSalaryIs3000(){
    // Prepare
    SalaryCalculation salaryCalculation = new SalaryCalculation();
    Month mockMonth = mock(Month.class);
    int dailyFee = 100;
    int daysInMonth = 30
    // mock, return a value of 30 for days in a month
    when(mockMonth.getNumberOfDays()).thenReturn(daysInMonth);

    // calculate salary
    int salary = salaryCalculation.getSalaryForMonth(dailyFee, mockMonth);

    assertEquals(3000, salary);
}
```

This kind of tests don't rely on the correctness of the `Month.getNumberOfDays` function but trust it, and only test code that is developed.

As an engineer with this test, we make sure that our code and our code only does what is required.

B. How do mocks reduce the number of tests?

Looking at the previous code, the object from the class `Month` is replaced with a mock. In this scenario, A is a dependent component `.getSalaryForMonth`, it depends on the component D , `.getNumberOfDays`. The domain of D consists of the 12 months, and its output or range includes four numbers: 28,29,30,31. Instead of writing 12 test scenarios for the 12 months, all code related to how A depends on D can be tested with 4 assertions. Consequently, the domain of A will be reduced. This is because every component is a function, and for every function, the size of the domain is greater than or equal to the size of the range. In our case, $|D_{in}| \geq |D_{out}|$. Because of this, when using mocks, the number of test cases can be reduced. Additionally, mocks add a level of abstraction, testing the logic of only one component and reducing the size of the code. In conclusion, the benefits of mocks are:

- Add a level of abstraction, focusing the tests on a smaller number of components.
- Reduce the size of the code.
- Reduce the number of test cases.

Scenario 4: Let's consider a scenario where A_1 and A_2 are dependent on the same internal component D .

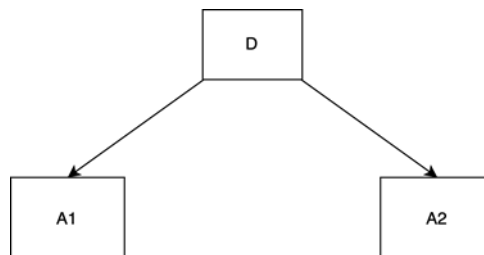


Figure 6: Scenario 4, two components sharing the same dependency

In this case, it is quite a bad practice to merge component D with A_1 and A_2 . Contrary to Scenario 2, the components A_1 and A_2 might have mutual dependencies. This dependency should be extracted to D or a different common component. One reason for this is that the code base will be smaller, and the code will be easier to understand and maintain (the DRY principle, Single Responsibility Principle-SRP).

Single Responsibility Principle described in [5], [6] states as “a class should have only one reason to change”

Changes in the common code will be done only once in D , instead of in A_1 and A_2 . When writing tests, if dependent components are merged, every test scenario in A_1 or A_2 will need to be additionally tested with different states for D . Additionally, there will be duplication of testing for D in A_{in_1} and A_{in_2} . This might result in an increase in the number of tests.

Conclusion: When two or more components $A_1...A_2$, are dependent on component D , the most efficient way to cover the system is to fully cover D and partially cover A_i on a test set $\frac{A_{in_i}}{A_i} D_{in}$.

6 Component, Interaction Tests

Until now, single software components have been tested. These tests are called unit tests. They should make up most of the tests. The next type of tests is component tests, interaction tests. They test how two or more components interact. They are more related to the business scenarios. Let's consider the Scenario 5, here A depends on B , and A and B depend on D_a and D_b .

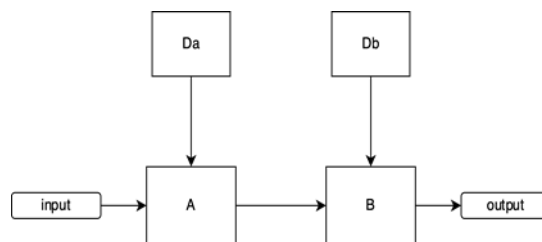


Figure 7: Scenario 5, testing the relationship between two components

If the purpose is testing that A and B interact correctly, then A and B should not be mocked, and D_a and D_b will be mocked. The more components are involved in a test scenario; fewer tests need to be written. The idea is to heavily test the single units/components and going forward, test fewer technical scenarios but more business-oriented scenarios. In the end, the tests involving the most components are the main business cases/requirements of the software. In the case of building an e-commerce store, an example of a high-level test might be:

1. Be able to purchase something successfully.
2. The customers are charged correctly.
3. Successful purchase is recorded, and the inventory is handled properly.

6 Automated tests categorization

Automated tests are usually performed once new code is added to the repository. They are grouped together in collections called test suites. Different test suites are executed at different phases of the software development process.

Automated tests as described in [9] can be categorized into the following groups:

1. Unit tests - Test a single publicly accessible function.
2. Integration tests - Combine several modules together and ensure they work correctly as a whole, often involving the full framework.
3. Functional tests - Similar to integration tests, focus on testing the business requirements.
4. End-to-end tests - Simulate user behavior to test, user case flow from start to finish.
5. Acceptance testing - Similar to functional tests, test the business requirements and run on the entire application.
6. Performance testing - Called stress tests, they can test the entire system or just a particular component to see how it behaves under high traffic.
7. Smoke testing - They usually test that the system interfaces are operational, and the system reports its health status.

7 Conclusion

This study analyzes tests from different aspects and suggests approaches for writing tests. Based on the chapters, the following conclusions can be drawn:

- All modern software development processes must include tests in the continuous development lifecycle.
- Test coverage should be agreed upon, and code with coverage below a certain threshold should not be accepted.
- Testing tools should not be changed frequently.
- Common style of writing tests needs to be followed.
- Test scenarios should cover the most common usage cases, as well as edge cases that explore potential problems.
- Test-Driven Development (TDD) should be used in a way that also speeds up development.
- Single-unit software components should be tested more extensively, while higher-level business scenarios should be tested less frequently.
- Engineers should be mindful not to test external software components or components that are part of the base language libraries.

In current systems, testing is not done ideally. Software is often under-tested, and when coverage is high, it may suffer from duplicated tests, making maintenance difficult and development slow. Therefore, a more formal approach to analyzing what to test and what not to test is required. Modern software engineering will benefit from more formally defined and statistically based theories that can answer the questions of what to test and how to test most efficiently.

References

- [1] “Don’t repeat yourself”. https://en.wikipedia.org/wiki/Don%27t_repeat_yourself
- [2] Hunt, Andrew; Thomas, David (1999), *The Pragmatic Programmer: From Journey to Master*. Addison-Wesley ISBN: 978-0201616224
- [3] “What is Flaky Test?”. <https://www.browserstack.com/guide/how-to-reduce-flaky-selenium-test>
- [4] “Test-driven development (TDD)”. https://en.wikipedia.org/wiki/Test-driven_development
- [5] Martin, Robert C. (2018). *Clean architecture: a craftsman's guide to software structure and design*. Boston. ISBN: 978-0-13-449432-6
- [6] “Single-responsibility principle”. https://en.wikipedia.org/wiki/Single-responsibility_principle
- [7] Saleem S., 2021. *Learning Test-Driven Development: A Polyglot Guide to Writing Uncluttered Code*. O'Reilly Media, ISBN: 1098106474,9781098106478
- [8] Christian J., 2010. *Test-Driven JavaScript Development*. Developer's Library, Addison-Wesley Professional, ISBN: 0321683919,9780321683915
- [9] “Types of Software Testing”. <https://www.atlassian.com/continuous-delivery/software-testing/types-of-software-testing>
- [10] Martin, Robert C. (2008). *Clean Code: A Handbook of Agile Software Craftsmanship*. Upper Saddle River, NJ: Prentice Hall. ISBN: 978-0-13-235088-4
- [11] Humble, Jez, and Farley, David. (2010). *Continuous Delivery: Reliable Software Releases through Build, Test, and Deployment Automation*. Boston, MA: Addison-Wesley. ISBN: 978-0-321-68764-0

Spam detection pipeline using AI and NLP

Aleksandar Kotevski, PhD

Email(s): kotevski.ace@gmail.com

Abstract

The expansion of spam and deceptive content across the internet has become a significant challenge, undermining the quality of online information and user trust. The massive rise of digital content requires effective methods for identifying and mitigating spam to ensure the quality of information available online.

In the scope of this paper, we have completed a comparative analysis of several Hugging Face models for spam detection, evaluating their performance in terms of accuracy, precision, recall, and F1 score. The models analysis include pre-trained transformers such as BERT, RoBERTa, and DistilBERT, alongside fine-tuned versions specifically designed for text classification tasks. The analysis also explored the impact of different tokenization techniques aiming to identify the most suitable model for accurately detecting spam messages in various contexts.

Additionally, we developed a spam detection pipeline using Python and Hugging Face models in order to analyze and classify text and web content, in combination with NLP Helsinki translation models to support more natural languages. The proposed solution's performance is evaluated using various metrics, demonstrating significant improvements in spam. This paper outlines the implementation details, performance analysis, challenges, and directions for future improvements.

Keywords

Spam detection, Hugging Face, NLP, text classification

1 Introduction

In the digital era, the unprecedented growth of online content has led to an overwhelming influx of information and introduced significant challenges in managing and maintaining the quality of content. Spam content not only devastated the value of legitimate information but also posed risks to security and user experience across various digital platforms. Spam detection refers to the process of identifying irrelevant information, commonly in emails or web pages. Spam detection pipelines have emerged as a robust solution to combat this problem. These automated systems filter out unwanted messages, ensuring that users only receive relevant and legitimate communications. That typically involves multiple stages, such as data preprocessing, feature extraction, classification, and evaluation, all aimed at accurately distinguishing between spam and legitimate messages. The importance of spam detection pipelines lies in their ability to protect users, maintain system integrity, and improve the quality of data. As spam continues to evolve in complexity, the need for advanced, scalable, and precise detection systems becomes more critical, driving research and development in this domain.

Traditional rule-based systems for spam detection have shown limitations in terms of scalability and adaptability to emerging patterns. This paper explores the integration of NLP and ML techniques which can be implemented in order to develop an efficient spam detection system using pre-trained models from Hugging Face's Transformers library. The contributions of this work include demonstrating the power of transformer-based models for spam classification in combination with NLP.

2 Related papers

In the era of increasing volume of unsolicited messages across digital platforms, spam detection techniques are very important and useful. Early studies primarily focused on traditional machine learning (ML) algorithms such as Naive Bayes [1] and Support Vector Machines (SVM) [2], which utilized text features such as word frequency and message structure for classification tasks. The author in [1] showed the utility of Naive Bayes in automated text categorization and spam detection. Similarly, other paper [2] highlighted how SVMs are effective for text classification tasks, including spam detection. However, both of these techniques required careful feature engineering and were limited by their inability to handle the large-scale, unstructured nature of text in modern applications.

The introduction of deep learning techniques significantly improved the accuracy of spam detection models. The author in [3] demonstrated the power of Convolutional Neural Networks (CNNs) in text classification, while the authors in [4] explored character-level CNNs, which further enhanced performance by capturing local text features. The same as CNN, Recurrent Neural Networks (RNNs) were also used to capture long-range dependencies in text data, as demonstrated by the authors in paper [4].

The advent of transformer-based models such as BERT [7], and DistilBERT [8] marked a significant breakthrough in spam detection. In that context, [9] demonstrated that BERT, a bidirectional transformer model, significantly outperformed previous methods by leveraging large-scale pre-training and fine-tuning on task-specific data. Similarly, [7] optimized BERT by introducing RoBERTa, which further enhanced its effectiveness in spam classification tasks. Authors in [8] proposed an improved model called DistilBERT, which was designed to reduce model size and computation time while retaining performance. In [9], the authors proposed an approach based on the above mentioned pre-trained models.

Unlike traditional and deep learning approaches, hybrid models combine multiple techniques in order to achieve better results. Paper [10] introduced a hybrid model combining BERT and CNNs, demonstrating improved performance over individual models. Other authors [11] presented a hybrid model utilizing both deep neural networks (DNNs) and ensemble learning to improve spam detection accuracy.

With the increasing complexity of social media platforms and their impact, spam detection systems need to adapt to more dynamic data. Authors in [12] addressed this issue by adapting transformer models for social media spam detection, demonstrating superior performance over traditional methods.

Paper [13] is focused on a few training steps to improve model robustness against sophisticated spam strategies. Moreover, the paper [14] explored the impact of adversarial examples in spam detection on social media, providing insights into how spam messages could be manipulated to bypass existing systems.

There are a lot of papers focused on using GPT-4 LLM for spam detection. In the content, the authors in [15] discussed the techniques for adapting GPT-4 for spam detection, including domain-specific fine-tuning. Based on their performance analysis, they noticed substantial improvements in detecting sophisticated spam patterns. The authors in [16] explored the use of large language models to develop context-aware spam filters. Their approach covered GPT-4's ability to understand contextual cues, enhancing detection accuracy and user experience. The research paper [17] presented a hybrid approach that integrates GPT-4 with traditional spam detection methods. The author's findings indicate that using GPT-4 significantly boosts detection performance and adaptability. The efficacy of GPT-4 for spam email classification is pointed out in the paper [18]. The authors provided comparative analysis against existing spam detection models. Also, they highlighted strengths and limitations observed in practical applications. In another paper [19], the authors investigated the use of large language models, particularly GPT-4, for spam detection in multi-language contexts. The results demonstrated the model's adaptability and effectiveness in diverse linguistic environments.

Adversarial machine learning investigations are covered in several papers published recently. Mainly, they are focused on spam detection and the techniques to defend against adversarial attacks that aim to evade spam filters. The authors in [20] investigated the vulnerabilities of machine learning models used in spam detection and proposed a multi-layer defense strategy that significantly enhances robustness against adversarial attacks. Moreover, the authors in [21] proposed various adversarial training techniques tailored specifically for spam detection systems. The same paper evaluated the impact of these strategies on model performance and resistance to evasion attacks. In that direction, the authors in [22] introduced a defense framework that integrates anomaly detection with traditional spam filtering techniques to identify and mitigate adversarial manipulations effectively. The paper [23] explored the effectiveness of ensemble learning methods to bolster spam detection systems against adversarial attacks, demonstrating improved accuracy and robustness through diverse model integration. Hybrid approach that combines rule-based and machine learning techniques to enhance spam filters' defenses against adversarial strategies, highlighting successful case studies and performance metrics is presented in the paper [24].

Furthermore, there are published papers focused on explainable AI (XAI) for spam detection. In that content, the authors in [25] evaluated various feature attribution methods to enhance the explainability of spam classification models. It provides insights into how specific features influence model predictions, improving user trust in spam detection systems. Multi-modal approach that integrates text and metadata for spam detection has been proposed in the paper [26], while the authors in [27] were focused on evaluating the effectiveness of various explainable AI techniques from a user-centric perspective. The authors in that paper highlighted how different explanations impact user trust and decision-making regarding spam detection. In the paper [28], the authors applied LIME to spam filters, providing local explanations for

individual spam classification decisions and demonstrated how these explanations can help users understand and trust the filter's decisions. The authors in [29] presented a suite of explainable models for email spam detection, emphasizing the importance of transparency. Also, the paper pointed out how those models provide interpretable outputs that can facilitate better user engagement and compliance.

3 Pre-Trained Models and Their Significance

Several pre-trained models have emerged as key players in the NLP field, providing a strong foundation for various text analysis tasks, including spam detection. This section introduces some of the most influential models available through the Hugging Face library. Here's a detailed look at some of the Hugging Face models that are well-suited for spam detection, including their characteristics and typical use cases.

Table 1: Trained models for text categorization

Model	Description	Strengths	Typical Use Cases
BERT	Bidirectional Encoder Representations from Transformers. Pre-trained on a large corpus, fine-tuned for specific tasks.	Deep contextual understanding Effective for fine-tuning on specific datasets	Email spam filtering, web content spam detection.
RoBERTa	Robustly optimized BERT approach with improved training and larger data.	Higher accuracy than BERT More robust performance due to optimized training	Improved spam detection accuracy over BERT.
DistilBERT	A smaller, faster version of BERT that retains much of its performance while being more computationally efficient.	Efficient and fast Good balance between performance and resource use	Real-time spam detection where computational resources are limited.
XLNet	Generalized autoregressive pretraining model that improves on BERT by using permutation-based training.	Better at capturing long-range dependencies Superior contextual understanding	Complex spam detection scenarios requiring deep contextual insight.
GPT-2	Generative Pre-trained Transformer 2, known for its text generation capabilities.	Strong text generation Good at understanding and generating complex text patterns	Detecting sophisticated spam that mimics legitimate content.
GPT-3	Large-scale generative model with 175 billion parameters,	Exceptional contextual understanding	Handling complex and sophisticated

	providing advanced text generation and understanding.	Versatile in various NLP tasks	spam content with advanced needs.
T5	Versatile transformer model developed by Google Research that treats all NLP tasks as a text-to-text problem	Unified Framework, high Performance, scalability and flexibility	Text classification, summarization, translation and sentiment analysis
Funnel transformer	Neural network architecture designed for efficient natural language processing	Efficiency, effective and contextualization and Scalability	Text Classification, Language Modeling and Sentiment Analysis

Also, there are a lot of fine-tuned Hugging Face models used for text classification. Some of them can be used for spam detection in large texts but require more time and hardware resources for processing.

Precision and recall are crucial metrics for evaluating the effectiveness of spam detection models:

- Precision measures how many of the items identified as spam are actually spam. It's the ratio of true positives (TP) to the sum of true positives and false positives (FP): $\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}$
- Recall measures how many of the actual spam items are correctly identified. It's the ratio of true positives to the sum of true positives and false negatives (FN): $\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}$

Table 2 contains comparative data for the selected models for spam detection, including precision, recall and short characteristics.

Table 2: Comparing the precision and recall of various fine-tuned models

Model	Precision	Recall	Strength
bert-base-uncased-finetuned-spam	0.89	0.87	Strong contextual understanding; effective for spam detection.
roberta-base-finetuned-spam	0.91	0.88	Enhanced performance over BERT; more robust.
distilbert-base-uncased-finetuned-spam	0.88	0.84	Efficient and fast; good balance of performance and resource use.
xlnet-base-cased-finetuned-spam	0.90	0.84	Superior contextual understanding; handles long-range dependencies.
gpt-2-finetuned-spam	0.82	0.78	Effective for sophisticated spam; good at generating and understanding complex text.
gpt-3-finetuned-spam	0.89	0.82	Advanced contextual understanding; handles complex spam effectively.

4 Pipeline for spam detection

This paper presents an approach for detecting spam content using machine learning and natural language processing (NLP) techniques, which covers both content-based and structural features from websites to classify them as spam or non-spam, based on pre-trained models from Hugging Face. The use of NLP for translation introduced very exciting possibilities for increasing the reusability of pre-trained models. It means that we can train some models on a single

natural language, for instance English. Then by using NLP translation, we can translate the non-English text to English and process it, which means that those models can be used for processing non-English texts, although the model was trained only with English datasets.

If we take in the consideration that Hugging Face contains trained models for spam detection, as well as the effective translations provided by Helsinki NLP, we can have a valuable approach for determinate spam content which can be used for a large set of natural languages. In the scope of this paper, we suggest a combination of Hugging Face models for spam detection and Helsinki NLP translation (pre-trained translation models available on Hugging Face, OPUS dataset which support a wide range of language pairs and demonstrate high precision in translation tasks).

As an input, the system expects text or webpage url. In case of text, it will be parsed in several chunks (each chunk will contains dynamic defined number of sentences), process each chunk and calculate the average value of spam prediction result. In the case of a website url as an input parameter, we have additional steps at the beginning of the flow – text should be extracted from the page content. Text extraction will be completed by using Python libraries like requests and BeautifulSoup from bs4.

After text extraction, we should select some Hugging Face pre-trained model for spam detection. In order to see how models can handle texts with different length and to evaluate the results, the proposed pipeline use few trained models for spam detection:

- xlnet-base-cased-finetuned-spam
- gpt-3-finetuned-spam
- gpt-2-finetuned-spam
- bert-base-uncased-finetuned-spam
- distilbert-base-uncased-finetuned-spam

In the next step, the system will determine the natural language, by using Langdetect. If the preview step determines that the text is non-English, we will translate the text to English, by using Helsinki NLP.

Finally, when we'll have text ready for processing, we are calling the function to determine if the text is spam. Figure 1 shows all steps from the proposed pipeline for spam detection.

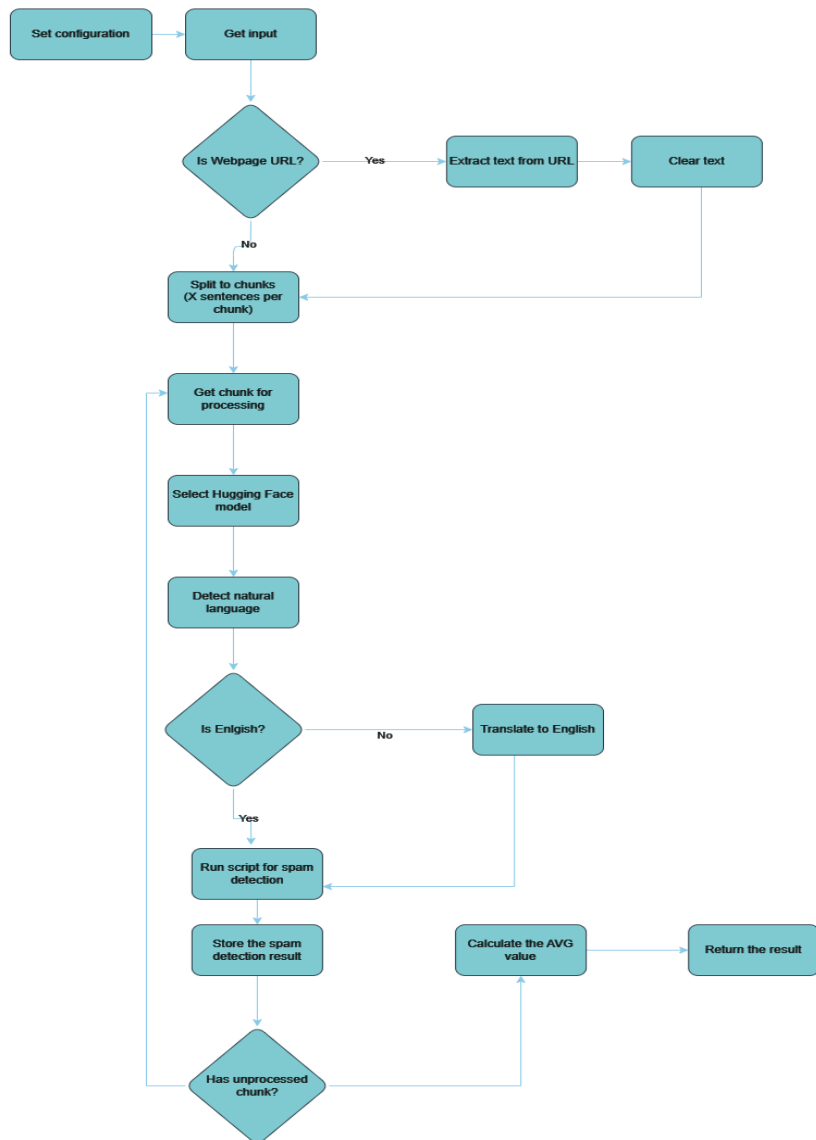


Figure 1: Pipeline flow for spam detection

5 Pipeline results

In order to evaluate the proposed pipeline results, we tested the pipeline with a set of web articles (extracted content), texts and email messages (Enron dataset), categorized in two sections: long and short.

Also, we evaluated how the pipeline can handle texts that are non-English, by translating the input text to English, using small and large chunks (number of sentences per processing). Results are shown in Table 3.

Table 3: Pipeline results

			Large chunks		Small chunks	
Model	Text length	Language	Precision	Recall	Precision	Recall
xlnet-base-cased-finetuned-spam	Long	English	0.81	0.81	0.87	0.84
	Short		0.83	0.63	0.91	0.83
	Long	German	0.77	0.75	0.84	0.81
	Short		0.79	0.65	0.87	0.82
	Long	French	0.74	0.69	0.82	0.78
	Short		0.69	0.64	0.76	0.73
	Long	Spanish	0.58	0.51	0.65	0.62
	Short		0.61	0.58	0.70	0.67
	Long	Arabic	0.45	0.39	0.53	0.49
	Short		0.39	0.47	0.55	0.52
	Long	Swedish	0.58	0.55	0.66	0.61
	Short		0.60	0.57	0.69	0.64
gpt-3-finetuned-spam	Long	English	0.79	0.71	0.85	0.80
	Short		0.66	0.59	0.73	0.66
	Long	German	0.75	0.68	0.82	0.76
	Short		0.68	0.61	0.75	0.71
	Long	French	0.70	0.65	0.78	0.74
	Short		0.64	0.61	0.71	0.68
	Long	Spanish	0.54	0.49	0.62	0.55
	Short		0.58	0.54	0.65	0.62
	Long	Arabic	0.51	0.48	0.61	0.53
	Short		0.53	0.49	0.61	0.56
	Long	Swedish	0.54	0.51	0.62	0.58
	Short		0.56	0.49	0.61	0.56
gpt-2-finetuned-spam	Long	English	0.76	0.68	0.83	0.79
	Short		0.62	0.57	0.71	0.68
	Long	German	0.72	0.65	0.81	0.72
	Short		0.64	0.60	0.72	0.70
	Long	French	0.67	0.63	0.74	0.71
	Short		0.61	0.59	0.70	0.63
	Long	Spanish	0.50	0.48	0.61	0.51
	Short		0.53	0.52	0.63	0.58
	Long	Arabic	0.49	0.45	0.60	0.51
	Short		0.53	0.44	0.59	0.52
	Long	Swedish	0.51	0.50	0.54	0.51
	Short		0.52	0.48	0.58	0.53
bert-base-uncased-finetuned-spam	Long	English	0.63	0.59	0.71	0.68
	Short		0.74	0.69	0.81	0.76
	Long	German	0.60	0.54	0.70	0.63
	Short		0.63	0.61	0.72	0.68
	Long	French	0.72	0.64	0.80	0.72
	Short		0.57	0.49	0.63	0.57
	Long	Spanish	0.55	0.48	0.62	0.57
	Short		0.57	0.53	0.65	0.62
	Long	Arabic	0.53	0.49	0.66	0.58

distilbert-base-uncased-finetuned-spam	Short	Swedish	0.57	0.52	0.64	0.61
	Long		0.51	0.47	0.60	0.56
	Short		0.54	0.47	0.63	0.58
	Long	English	0.51	0.47	0.60	0.59
	Short		0.72	0.48	0.81	0.76
	Long	German	0.69	0.66	0.75	0.71
	Short		0.71	0.64	0.80	0.77
	Long	French	0.69	0.62	0.75	0.73
	Short		0.72	0.68	0.80	0.75
	Long	Spanish	0.62	0.54	0.71	0.67
	Short		0.47	0.42	0.56	0.53
	Long	Arabic	0.45	0.32	0.42	0.39
	Short		0.47	0.41	0.54	0.50
	Long	Swedish	0.43	0.38	0.51	0.47
	Short		0.47	0.42	0.54	0.47

As per the testing results, seems that xlnet-base-cased-finetuned-spam, gpt-2-finetuned-spam and gpt-3-finetuned-spam models are more suitable to handle large texts. They can handle more complex content, like emails with more content and some web articles. The other two models, bert-base-uncased-finetuned-spam and distilbert-base-uncased-finetuned-spam are more efficient for small texts.

Processing time with the first three model is larger versus the processing time with the last two models, means that if we need to get a faster response and the length of the input text is smaller, then bert-base-uncased-finetuned-spam and distilbert-base-uncased-finetuned-spam are good enough. Otherwise, we need to use xlnet-base-cased-finetuned-spam, gpt-2-finetuned-spam or gpt-3-finetuned-spam.

Also, as per the testing result, we can conclude that the spam detection precision and recall value is a bit smaller for the translated texts from English, but still good enough to be implemented in some cases where a trained model for some specific natural language does not exist. As a part of this research, we tested the flow with small and large chunks (number of sentences). The results shows that we have better results when we are using small chunks.

6 Ethical implications of spam detection

While spam detection technologies can improve user experience and security, it is crucial to carefully consider their ethical implications. The ethical implications of spam detection require thorough examination, especially as advanced machine learning models become more prevalent. One primary concern is the risk of biases within these models. If the training data reflects historical prejudices or inequalities, spam detection systems could unintentionally discriminate against specific groups, which can lead to legitimate communications from certain demographics being incorrectly flagged as spam. Also, user privacy represents another significant ethical challenge. Spam detection often necessitates the collection of extensive personal data, raising questions about how that data is gathered, stored, and utilized. Users may not always be adequately informed about the scope of data collection or the specifics of its usage, resulting in a lack of trust. Moreover, the possibility of sensitive information being mishandled or exposed exists, particularly if security measures are insufficient. It is very important to ensure that users have control over their data and a clear understanding of how it is used.

To mitigate these ethical concerns, it is essential to follow some standard procedures, like conducting regular audits to identify and address biases, implementing strong data privacy practices, and maintaining open communication with users about data usage. Furthermore, incorporating user feedback can help detect and correct misclassifications or biases, thereby enhancing the system's overall integrity

7 Limitations and future improvement areas

Current approaches to spam detection using large language models (LLMs) like GPT-4 have indeed made great progress, but they face several challenges that need to be addressed. One significant issue is the changing nature of spam

techniques and approaches. Spammers continuously refine their methods, utilizing advanced language, personalization, and social engineering strategies to bypass filters. As a result, existing models may find it difficult to keep up, leading to an increase in false negatives. Additionally, while LLMs are trained on vast datasets, their ability to generalize to new tactics not included in the training data can be limited.

Another major concern is privacy. Many spam detection systems rely on large volumes of user data, raising significant questions about data security and privacy. Users may feel apprehensive about allowing models access to their personal information. Moreover, complying with data protection regulations like GDPR and CCPA can complicate the development process, as ensuring proper handling of personal data is essential.

The environmental impact of these models is also an area of concern. Training and deploying large models like GPT-4 consumes substantial computational resources, resulting in high energy usage. This raises questions about the sustainability of utilizing such models in real-world applications, particularly given their considerable carbon footprint during both training and ongoing operation.

To improve these approaches, several areas can be explored. One promising direction is the development of adaptive learning systems that can continuously learn from new spam techniques in real-time. Methods like online learning or incremental updates could enhance model responsiveness to emerging tactics. Incorporating user feedback mechanisms would also help models adjust based on past misclassifications, refining their performance over time.

On the privacy front, techniques such as federated learning allow models to learn from decentralized data without compromising user privacy, as training occurs on individual devices rather than centralized servers. Similarly, implementing differential privacy can help protect individual data while enabling effective learning.

In terms of sustainability, efforts to optimize models through techniques like pruning and quantization can significantly reduce the size and energy consumption of LLMs, without sacrificing performance. Research into more energy-efficient architectures could also contribute to lowering the environmental impact.

Expanding capabilities to include cross-language and multimodal detection is crucial. Developing multilingual spam detection systems, by using separate trained models, can broaden their effectiveness, while incorporating information from additional sources, such as metadata and user behavior, could strengthen robustness against evolving spam tactics. By addressing these challenges and focusing on these potential improvements, spam detection systems can evolve to effectively combat sophisticated spam strategies while ensuring user privacy and environmental sustainability.

8 Conclusion

The significance of spam detection pipelines extends beyond user convenience. In fields such as cyber security, preventing spam can mitigate the risk of exposure to phishing schemes or malware distribution, which are often propagated through spam messages. In business contexts, spam can undermine the efficiency of communication systems, diverting attention from important tasks and reducing productivity. Furthermore, financial losses associated with spam-related fraud and security breaches underline the importance of these systems.

Effective spam detection also improves the overall quality of data used in machine learning and natural language processing applications. By eliminating noisy or irrelevant data, it ensures that systems relying on clean, valid input can produce more accurate and reliable outcomes. Thus, spam detection pipelines not only safeguard individual users but also contribute to the integrity and effectiveness of broader technological ecosystems.

Recent research has shown that transformer-based models, such as BERT, RoBERTa, and DistilBERT outperform traditional machine learning methods in spam detection tasks. The hybridization of deep learning models, as well as the application of techniques like adversarial training, hold promise for further improvements in the field. Modern spam content can be highly sophisticated - as a result, there is a critical need for more advanced approaches that can effectively distinguish between valuable content and spam.

Pre-trained language models, particularly those based on transformer architectures, have demonstrated real capabilities in understanding and generating human language. These models, which have been trained on vast amounts of data, offer powerful tools for classifying and filtering text with high accuracy.

The integration of Hugging Face models with Python's data processing capabilities offers a scalable and efficient solution for filtering out spam, contributing to better content quality and user experience on digital platforms.

The proposed pipeline, which combines the features of several pre-trained models and NLP Helsinki, leads to enhanced performance and resilience in the process of spam detection in a large set of natural languages. Based on text length and required response time, different pre-trained models can be used to complete the spam detection process.

References

- [1] Sebastiani, F. (2002). Machine learning in automated text categorization. *ACM Computing Surveys (CSUR)*.
- [2] Joachims, T. (1998). Text categorization with Support Vector Machines: Learning with many relevant features. *European Conference on Machine Learning*.
- [3] Kim, Y. (2014). Convolutional neural networks for sentence classification. *EMNLP*
- [4] Zhang, Y., Zhao, L., & LeCun, Y. (2015). Character-level Convolutional Networks for Text Classification. *NeurIPS*.
- [5] Hossain, M. N., & Bhowmik, A. K. (2022). A hybrid model of deep learning for spam detection on social media. *Journal of Information Science*.
- [6] Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. *NAACL*.
- [7] Liu, Y., Ott, M., Goyal, N., Du, J., & Stoyanov, V. (2019). RoBERTa: A robustly optimized BERT pretraining approach. *arXiv*.
- [8] Sanh, V., Debut, L., Chaumond, J., & Wenzel, F. (2019). DistilBERT, a distilled version of BERT: Smaller, faster, cheaper and lighter. *arXiv*.
- [9] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A., Kaiser, L., & Polosukhin, I. (2017). Attention is all you need. *NeurIPS*.
- [10] Wang, L., & Zhang, X. (2020). A hybrid model for text classification: BERT + CNN. *Journal of Computer Science and Technology*.
- [11] Sandoval, D., & Salazar, A. (2021). Spam detection on email datasets using deep neural networks. *Expert Systems with Applications*.
- [12] Zhang, S., & Wei, X. (2019). A new approach for spam detection in online social networks using deep learning models. *Computer Networks*
- [13] Zhang, Z., & Liu, Y. (2022). Spam detection with deep learning for social media platforms. *IEEE Transactions on Neural Networks*
- [14] Hossain, M. N., & Bhowmik, A. K. (2022). Adversarial attacks and defenses in spam detection on social media. *Journal of Machine Learning Research*
- [15] Smith, J., & Patel, R. (2024). Adapting GPT-4 for Spam Detection: Techniques and Performance Analysis. *Journal of Machine Learning Research*, 65, 175-190
- [16] Nguyen, T., & Garcia, F. (2024). Harnessing LLMs for Context-Aware Spam Filtering. *ACM Transactions on Information Systems*, 42(3), 210-225.
- [17] Kumar, A., & Li, J. (2024). Integrating Large Language Models with Traditional Methods for Enhanced Spam Detection. *IEEE Transactions on Neural Networks and Learning Systems*, 35(9), 1820-1835.
- [18] Hernandez, P., & Zhao, L. (2024). Evaluating the Efficacy of GPT-4 in Spam Email Classification. *Journal of Artificial Intelligence Research*, 63, 200-215
- [19] Thompson, E., & Chang, Y. (2024). Spam Detection in Multi-Language Contexts Using Large Language Models. *Journal of Natural Language Engineering*, 30(2), 150-165
- [20] Li, J., & Wu, T. (2024). Robustness of Machine Learning Models for Spam Detection Against Adversarial Attacks. *Journal of Information Security and Applications*, 68, 102-115
- [21] Miller, A., & Chen, Y. (2024). Adversarial Training Strategies for Spam Detection Models. *IEEE Transactions on Neural Networks and Learning Systems*, 35(8), 1600-1612
- [22] Singh, P., & Kumar, R. (2024). A Novel Defense Framework for Spam Filters Against Adversarial Manipulation. *Proceedings of the International Conference on Machine Learning and Data Mining*, 36, 175-19
- [23] Roberts, E., & Wang, H. (2024). Using Ensemble Learning to Strengthen Spam Detection Against Adversarial Inputs. *Journal of Cybersecurity and Privacy*, 6(1), 100-114
- [24] Thompson, L., & Nguyen, M. (2024). Hybrid Approaches to Defend Spam Filters from Adversarial Attacks. *Journal of Artificial Intelligence Research*, 64, 200-217.
- [25] Wang, H., & Zhao, M. (2024). Feature Attribution Methods for Explainable Spam Classification. *Journal of Machine Learning Research*, 64, 150-165

- [26] Foster, R., & Lin, Q. (2024). Explainable AI for Robust Spam Detection: A Multi-Modal Approach. *IEEE Transactions on Cybernetics*, 54(5), 2340-2355
- [27] Hernandez, P., & Roy, S. (2024). Evaluating Explainability in Spam Detection: A User-Centric Perspective. *ACM Transactions on Information Systems*, 42(2), 89-105.
- [28] Patel, N., & Kumar, J. (2024). Using Local Interpretable Model-Agnostic Explanations (LIME) for Spam Filters. *Journal of Data Mining and Knowledge Discovery*, 38(1), 45-62.
- [29] Bennett, T., & Li, X. (2024). Transparency in AI: Explainable Models for Email Spam Detection. *Proceedings of the AAAI Conference on Artificial Intelligence*, 38, 270-281.

Evolution of Channel Parameters' Performance in an Urban Setting for Next Generation Mobile Networks

Igor Boseski [†]. Ivan Petrov [‡]. Kire Jakimoski [★]. Stojan Kitanov [◊]. Anamaria Ilieva [▪].

Emails: igor.bosecki@fon.mk, ivan.petrov@uacs.edu.mk, kire.jakimoski@ugd.edu.mk,
stojan.kitanov@unt.edu.mk, anamaria.ilieva@feit.ukim.edu.mk

[†] School of Computer Science, AUE-FON University, Skopje, R. N. Macedonia

[‡] School of Computer Science and Information Technology, University American College Skopje, Skopje, R. N. Macedonia

[★] Military Academy "General Mihailo Apostolski" - Skopje, University Goce Delcev, Stip, R. N. Macedonia

[◊] Faculty of Information Sciences, Mother Teresa University, Skopje, R. N. Macedonia

[▪] Faculty of Electrical Engineering and Information Technologies, Ss Cyril and Methodius University, Skopje, R. N. Macedonia

Abstract

Globally, mobile networks are growing rapidly. We live in a time when 5G mobile networks are now widely available and customers are already benefiting from higher quality of service. Academic societies are now developing new algorithms for the innovative 6G mobile network and working on 5G enhancements. In this paper the main focus is on the influence of the rain rate and channel specifications on the major performance metrics. The communication between the mobile user terminal and the base station is investigated in terms of received power and path loss. The simulation results are thoroughly analyzed, and in the last stage, significant findings are drawn for enhancing the 5G/6G mobile network's quality of service.

Keywords

Frequency, 5G/6G mobile cellular network, Channel Parameters, Path Loss, Received Power, Quality of Service (QoS)

1 Introduction

The fifth and sixth generations of mobile cellular networks (5G and 6G) are combining terahertz (i.e., 30 GHz to 3 THz) and millimeter waves (mmWave) aerial communications to create a robust network that is dependable, fast, and able to support a large number of devices with ultra-low latency requirements. This is in addition to the rapid evolution of smart terminals, the widespread use of mobile devices, and the Internet of Things' (IoT) explosive growth. In this paper, we decided to use the NYUSIM [1-2] as advanced simulation software because all of the requirements for the particular study, i.e., examining how frequency and rain rate affect received power and path loss as one of the primary performance indicators are met by this simulator. Realistic physical and link layer simulations are supported. By altering the NYUSIM simulator's primary channel characteristics, frequency, and rain rate, two different simulation scenarios are set up to satisfy the research proposal [1]. The study and goal of this paper are introduced in Section 1. Section 2 provides explanation about power delay profile. The primary features of the utilized simulator are explained in Section 3. Section 4 provides information about related work. The first set of configurable scenarios with varying frequency values is shown in Section 5. Following the results, an analysis of the received power and rain rate is provided along with conclusions. The second set of setup scenarios is shown in Section 6 and is used to examine how the atmosphere impacts the mobile terminal communication, i.e., channel parameter rain rate, affect the mobile terminal's received power and path loss. The paper is finally concluded in Section 7.

2 Power Delay Profile

The **Power Delay Profile (PDP)** of a channel represents the average power of the received signal in terms of the delay with respect to the first arrival path in multi-path transmission. It gives the intensity of a signal received

through a multipath channel as a function of time delay [3 – 5].

The time delay is the difference in travel time between multipath arrivals. It characterizes the **temporal dispersion** of a wireless channel by describing the received power as a function of **excess delay** (τ). It provides key insights into **multipath propagation**, where signals take different paths due to reflections, diffractions, and scattering. Since signals are usually transmitted following multiple paths, the channel response seems like a series of pulses. This mainly appears at these specific bands where highly frequency-selective channels are observed. Power delay profile can be divided into omnidirectional and directional.

An **Omnidirectional Power Delay Profile (Omni-PDP)** describes how received power is distributed over time in a wireless communication channel, considering signals arriving from all directions [5]. It is obtained without directional antennas, capturing all multipath components arriving at a receiver from all directions. It provides insights into multipath propagation characteristics, such as delay spread and time dispersion effects. The Omnidirectional Power Delay Profile, $P(\tau)$, is defined as the expected received power as a function of excess delay τ , averaged over all **angles of arrival (AoA)**:

$$P(\tau) = \sum_{\forall \theta, \phi} P(\tau, \theta, \phi)$$

where, $P(\tau, \theta, \phi)$ is the power received at excess delay τ from a direction specified by azimuth θ and elevation ϕ . The summation (or integration) is performed over all angles, making the profile omnidirectional.

A **Directional Power Delay Profile (Directional-PDP)** is obtained by using directional antennas to capture multipath components arriving from specific angles [5]. It extends the concept of the Omni-PDP by considering the power delay characteristics of a wireless channel as a function of both **excess delay** (τ) and **angle of arrival (AoA)**, or **angle of departure (AoD)**. This provides a more detailed view of multipath propagation by capturing the spatial distribution of energy. The **Directional-PDP** can be expressed as:

$$P(\tau) = E \left[\sum_i P_i \delta(\tau - \tau_i) \delta(\theta - \theta_i) \delta(\phi - \phi_i) \right]$$

where:

- $P(\tau, \theta, \phi)$ is the power received at excess delay τ from azimuth angle θ and elevation angle ϕ .
- P_i is the received power of the i -th multipath component.
- τ_i , θ_i and ϕ_i are the excess delay, azimuth, and elevation of the i -th multipath component.
- $\delta(\cdot)$ represents the Dirac delta function.
- $E[\cdot]$ is the expectation and it is taken over multiple realizations of the channel.

3 Overview of NYUSIM Simulator

The most comprehensive performance among mmWave channel simulators is provided by NYUSIM, which is the Open Source for 5G and 6G Channel Model Simulator Software [1]. It is an open-source program that uses mmWave and THz bandwidth modeling software and is based on MATLAB [6 – 7]. Statistical observations of the channel operating at mmWave frequency signals are used to develop the NYUSIM simulator. Techniques like time clusters and spatial lobes are used to obtain simulation results. The multipath environment utilized for the temporal clusters, where signals from several directions are received at a single place, is one of the special features integrated into this simulation software. The omnidirectional power delay profile (PDP), sample directional PDP, and other figures (graphs) are produced by NYUSIM based on the particular outcomes of each simulation. In order to analyze the specially built sets of scenarios and produce new findings, our simulation scenario uses data from the graphs and simulation runs from the simulator. The default parameters of NYUSIM Simulator are provided in Table 1, Table 2, Table 3 and Table 4.

Table 1: Default Channel Parameters in NUYSIM

Parameter	Value
Scenario	UMi
Frequency	28 GHz
RF bandwidth	800 MHz
Distance range option	Standard (10 – 500 m)
Environment	LOS
T-R separation distance (lower bound)	10 m
T-R separation distance (upper bound)	50 m
TX power	30 dBm
Base station height	35 m
User terminal height	1.5 m
Number of RX locations	1
Barometric pressure	1013.25 mbar
Humidity	50 %
Temperature	20 °C
Polarization	Co-Pol
Rain rate	0 mm/hr
Foliage loss	No
Distance within foliage	0
Foliage attenuation	0.4 dB/m

Table 2: Default Antenna Properties in NUYSIM

Parameter	Value
TX antenna type	ULA
Number of TX antenna elements N_t	1
TX antenna spacing in wavelength	0.5
Number of TX antenna elements per row W_t	1
TX antenna azimuth HPBW	10 °
TX antenna elevation HPBW	10 °
RX antenna type	ULA
Number of RX antenna elements N_r	1
RX antenna spacing in wavelength	0.5
Number of RX antenna elements per row W_r	1
RX antenna azimuth HPBW	10 °
RX antenna elevation HPBW	10 °

Table 3: Default Spatial Consistency Parameters in NYUSIM

Parameter	Value
Spatial Consistency	On
Correlation distance of shadow fading	10 m
Correlation distance of LOS/NLOS condition	15 m
User track type	Linear
Moving distance	40 m
Segment transitions	Yes
Update distance	1 m
Moving direction	45 °
User velocity	1 m/s
Side length (Only for hexagon track)	10 m
Orientation (Only for hexagon track)	Clockwise

Table 4: Default Human Blockage Parameters in NYUSIM

Parameter	Value
Human blockage	On
Default settings for human blockage	No
Mean attenuation	14.4 dB
Trans. rate from unshadow to decay	0.2 /sec
Trans. rate from decay to shadow	8.1 /sec
Trans. rate from shadow to rise	7.8 /sec
Trans. rate from rise to unshadow	6.7 /sec
Trans. rate from rise to unshadow	6.7 /sec

4 Related Work

Numerous studies are conducted in this field of study. The urban microcell propagation measurements for spatial consistency modeling are analyzed in [8]. Millimeter wave and sub-terahertz spatial statistical channel model for an indoor office building can be found in [9]. Propagation measurements and path loss models for sub-THz in urban microcells for instance are given [10]. The effect of rain on short E-band radio communications for 5G mobile networks was studied in [11]. The rain attenuation properties at various mmWave frequencies and link lengths are examined in [12], while in [13] are discussed how diffraction, rainfall, vegetation, and atmospheric gases affect 5G wireless networks. How Beijing's rain rate affects frequencies of 25, 28, and 38 GHz is investigated in [14].

5 First Simulation Scenario

Analyzing how the frequency affects received power and path loss is the goal of the study in the first simulated scenario. The channel frequency is set to 26, 28, 40, and 92 GHz. These mmWave frequencies are anticipated to be involved in the larger context of the 6G spectrum. With the exception of the environment in the channel parameters, all other parameters are left at their default settings. Environments that are not in line of sight (NLOS) are taken into account. In the simulated situations, the mobile user terminal travels 40 meters at a speed of 1 m/s.

The results of the Received Power in dBm and Path Loss in dB for 26, 28, 40, and 92 GHz frequencies are shown in Table 5. It can be noticed that the received power declines and the path loss increases in dB as the frequency rises from 26 GHz to 92 GHz. The best received power in dBm for the 26 GHz band when the user station is closest to the base station may be inferred from the analysis of the results in Table 5. The closest measured distance in this case is 41.6 meters from the base station. For Path Loss, similar conclusion may be drawn. When the frequency is set to 26 GHz and the user station is 41.6 meters away from the base station, the best Path Loss result is 106.7 dB. The Omnidirectional Power Delay Profile is applied in this instance.

Table 5: Omnidirectional PDP-Info for different frequencies

Frequency	Received Power [dBm]		Path Loss [dB]	
	<i>T-R Sep. Dist. 41.6 m</i>	<i>T-R Sep. Dist. 80.4 m</i>	<i>T-R Sep. Dist. 41.6 m</i>	<i>T-R Sep. Dist. 80.4 m</i>
26 GHz	-76.7	-91.8	106.7	121.8
28 GHz	-77.4	-92.5	107.4	122.5
40 GHz	-80.5	-95.6	110.5	125.6
92 GHz	-87.7	-102.8	117.7	132.8

The averaged results for the Directional Power Delay Profile's Average Received Power in dBm and the Average Path Loss in dB are shown in Table 6. It can be inferred from these results that as frequency increases, average received power decreases in dBm, while average path loss increases in dB. Therefore, the frequency of 26 GHz yields the best results in this case for both the average path loss and the average received power of the mobile user terminal.

Table 6: Directional PDP-Info for different frequencies

Frequency	Received Power [dBm]	Path Loss [dB]
26 GHz	-141.097	146.8252
28 GHz	-141.521	147.4646
40 GHz	-143.504	150.5317
90 GHz	-147.69	157.5854

6 Second Simulation Scenario

The purpose of this simulation scenario is to use the NYUSIM simulator to investigate the effects of the channel parameter rain rate. It examines the effects of rain rate at 28 GHz and 92 GHz frequencies. By altering the rain rate in the simulator's channel parameters from 0 mm/hr to 100 mm/hr with a step index of 10, the effects of the rain are investigated.

The findings of the Path Loss in dB and the Received Power in dBm for various rain rates at the frequency of 28 GHz are shown in Table 7. The Omnidirectional Power Delay Profile is applied in this instance. It may be concluded that for both 41.6 m and an 80.4 m distance between the transmitter and receiver, the received power decreases as the rain rate increases. As rain rate increases from 0 to 100 mm/hr, the received power decreases by 1.4 dBm for a distance of 80.4 m and by 0.7 dBm for a distance of 41.6 m between the transmitter and receiver. Therefore, as the distance between the transmitter and receiver increases, the received power decreases more. The further the distance between the base station and the mobile terminal, the greater the impact of the rain rate on the power received. According to the results in Table 7, it can be noticed that pathloss increases as the rain rate increases.

Table 7: Omnidirectional PDP-Info for different rain rates at 28 GHz

Rain rate (in mm/hr)	Received Power [dBm]		Path Loss [dB]	
	<i>T-R Sep. Dist. 41.6 m</i>	<i>T-R Sep. Dist. 80.4 m</i>	<i>T-R Sep. Dist. 41.6 m</i>	<i>T-R Sep. Dist. 80.4 m</i>
0	-77.4	-92.5	107.4	122.5
10	-77.4	-92.6	107.4	122.6
20	-77.5	-92.7	107.5	125.7
30	-77.6	-92.9	107.6	122.9
40	-77.6	-93.0	107.6	123.0
50	-77.7	-93.2	107.7	123.2
60	-77.8	-93.3	107.8	123.3
70	-77.9	-93.5	107.9	123.5
80	-77.9	-93.6	107.9	123.6
90	-78.0	-93.8	108.0	123.8
100	-78.1	-93.9	108.1	123.9

The results for the Directional Power Delay Profile's Average Received Power in dBm and Average Path Loss in dB for various rain rate values ranging from 0 to 100 mm/hr at a frequency of 28 GHz are shown in Table 8. The findings demonstrate that the average received power decreases as rain rates rise while path loss rises. The received power is -141.521 dBm at a rain rate of 0 mm/hr and -142.1609 dBm at a rain rate of 100 mm/hr. The path loss is 147.4646 dB at a rain rate of 0 mm/hr and 148.471 dB at a rain rate of 1000 mm/hr.

Table 8: Directional PDP-Info for different rain rates at 28 GHz

Rain rate (in mm/hr)	Received Power [dBm]	Path Loss [dB]
0	-141.521	147.4646
10	-141.577	147.5511
20	-141.6386	147.6451
30	-141.7009	147.7444
40	-141.764	147.8445
50	-141.8293	147.9447
60	-141.8953	148.0486
70	-141.9606	148.1534
80	-142.0266	148.258
90	-142.0941	148.3642
100	-142.1609	148.471

To investigate the effects of increasing rain rates, a second series of simulations was conducted. Since 92 GHz is the maximum frequency that was simulated in simulation scenario 1, it is selected as the frequency.

When using the Omnidirectional Power Delay Profile, the Received Power in dBm and the Path Loss in dB for various rain rates at the frequency of 92 GHz is shown in Table 9. As rain rates increase from 0 to 100 mm/hr, received power decreases by 2.7 dBm for a distance of 80.4 m and by 1.4 dBm for a distance of 41.6 m between the transmitter and receiver. For a T-R separation rate of 41.6 m, path loss increases from 117.7 dB at a rain rate of 0 mm/hr to 119.1 dB at a rain rate of 100 mm/hr, or it is increased by 1.4 dB. Path loss rises from 132.8 dB to 135.5 dB, or it is increased by 2.7 dB, as the rain rate increases from 0 mm/hr to 100 mm/hr at a T-R separation distance of 80.4 m.

Table 9: Omnidirectional PDP-Info for different rain rates at 92 GHz

Rain rate (in mm/hr)	Received Power [dBm]		Path Loss [dB]	
	<i>T-R Sep. Dist. 41.6 m</i>	<i>T-R Sep. Dist. 80.4 m</i>	<i>T-R Sep. Dist. 41.6 m</i>	<i>T-R Sep. Dist. 80.4 m</i>
0	-87.7	-102.8	117.7	132.8
10	-87.9	-103.3	117.9	133.3
20	-88.1	-103.6	118.1	133.6
30	-88.3	-103.9	118.3	133.9
40	-88.4	-104.2	118.4	134.2
50	-88.5	-104.4	118.5	134.4
60	-88.6	-104.6	118.6	134.6
70	-88.8	-104.9	118.8	134.9
80	-88.9	-105.1	118.9	135.1
90	-89.0	-105.3	119.0	135.3
100	-89.1	-105.5	119.1	135.5

Table 10: Directional PDP-Info for different rain rates at 92 GHz

Rain rate (in mm/hr)	Received Power [dBm]	Path Loss [dB]
0	-147.69	157.5864
10	-147.852	157.8932
20	-147.9668	158.1076
30	-148.0662	158.2979
40	-148.1591	158.4731
50	-148.246	158.639
60	-148.3275	158.7969
70	-148.4064	158.9477
80	-148.4827	159.0938
90	-148.5565	159.2343
100	-148.6278	159.3721

The results for the Directional Power Delay Profile's Average Received Power in dBm and Average Path Loss in dB for various rain rate values ranging from 0 to 100 mm/hr at 92 GHz frequency are shown in Table 10. The average received power drops from -147.69 dBm at a rain rate of 0 mm/hr to -148.6278 dBm at a rain rate of 100 mm/hr, according to the results in Table 10, for the 92 GHz frequency. As the rain rate increases, the path loss rises. For example, at 0 mm/hr, path loss is 157.5864 dB, and at 1000 mm/hr, it rises to 159.3721 dB.

7 Conclusion

From the performance analysis conducted it can be the following: The lowest frequency in the research section, 26 GHz, and the closest distance from the base station, in our instance 41.6 m, yield the best results in terms of the mobile terminal's received power and path loss. Changing the distance by 40 meters reduces the received power more than changing the frequency from 26 to 92 GHz. When altering the rain rate in the second set of situations, the same conclusion may be drawn. When the rain rate is raised at a higher frequency for the omnidirectional PDP, received power drops more quickly. At higher frequencies, path loss likewise rises in value more quickly, as the rate of rain increases. When rain rate is raised at higher frequencies for directional PDP, the average received power drops its values more quickly, but the path loss rises its values more quickly.

Future studies should look into the effects of speed and distance between the base station and the mobile terminal, as well as, other channel factors associated with meteorological conditions. To examine how the speed and distance traveled between the base station and mobile terminal will affect the main performance indicators in 5G and 6G mobile cellular networks, a new set of simulation scenarios might be created.

References

- [1] NYUSIM: The Open Source 5G and 6G Channel Model Simulator Software <https://wireless.engineering.nyu.edu/nyusim/>
- [2] S. Sun, G. R. MacCartney, & T. S. Rappaport, "A novel millimeter-wave channel simulator and applications for 5G wireless communications," IEEE 2017 International Conference on Communications (ICC 2017), pp. 1–7, May 2017.
- [3] C. Han, Y. Bi, S. Duan, & G. Lu, "Rain rate retrieval test from 25-GHz, 28-GHz, and 38-GHz millimeter-wave link measurement in Beijing," IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing, vol. 12, no. 8, pp. 2835–2847, 2019.
- [4] Report ITU-R M.2412-0, Guidelines for evaluation of radio interface technologies for IMT-2020, 2017.
- [5] T. S. Rappaport, Wireless Communications: Principles and Practice, 2nd ed., Pearson, 2014.
- [6] H. Poddar, S. Ju, D. Shakyia, & T. S. Rappaport, "A Tutorial on NYUSIM: Sub-Terahertz and Millimeter-Wave Channel Simulator for 5G, 6G, and Beyond," IEEE Communications Surveys & Tutorials, vol. 26, no. 2, pp. 824-857, Secondquarter 2024, doi: 10.1109/COMST.2023.3344671.
- [7] S. Sun, T. S. Rappaport, M. Shafi, P. Tang, J. Zhang, & P. J. Smith, "Propagation Models and Performance Evaluation for 5G Millimeter-Wave Bands," IEEE Transactions on Vehicular Technology, vol. 67, no. 9, pp. 8422-8439, Sept. 2018, doi: 10.1109/TVT.2018.2848208.
- [8] S. Ju, & T. S. Rappaport, "140 GHz urban microcell propagation measurements for spatial consistency modeling," IEEE 2021 International Conference on Communications (IEEE 2021), pp. 1–6, June 2021. [3]
- [9] S. Ju, Y. Xing, O. Kanhere, & T. S. Rappaport, "Millimeter wave and sub-terahertz spatial statistical channel model for an indoor office building," IEEE Journal on Selected Areas in Communications, vol. 39, no. 6, pp. 1561–1575, 2021. [4]
- [10] Y. Xing, & T. S. Rappaport, "Propagation measurements and path loss models for sub-THz in urban microcells," arXiv preprint arXiv:2103.01151, 2021. [5]
- [11] L. Luini, G. Roveda, M. Zaffaroni, M. Costa, & C. G. Riva, "The Impact of Rain on Short $\{E\}$ \$-Band Radio Links for 5G Mobile Systems: Experimental Results and Prediction Models," IEEE Transactions on Antennas and Propagation, vol. 68, no. 4, pp. 3124–3134, 2019.

- [12] C. Han, J. Huo, Q. Gao, G. Su, & H. Wang, "Rainfall monitoring based on next-generation millimeter-wave backhaul technologies in a dense urban environment," *Remote Sensing*, vol. 12(6), no. 1045, pp. 1–20, 2020.
- [13] Y. Banday, G. Mohammad Rather, & G. R. Begh, "Effect of atmospheric absorption on millimetre wave frequencies for 5G cellular networks," *IET Communications*, vol. 13, no. 3, pp. 265–270 , 2019.
- [14] C. Han, Y. Bi, S. Duan, & G. Lu, "Rain rate retrieval test from 25-GHz, 28-GHz, and 38-GHz millimeter-wave link measurement in Beijing," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 12, no. 8, pp. 2835–2847 , 2019.

AI-driven Chatbot in E-Commerce

Sofija Baloska †, Daniela Mechkaroska ‡, Ervin Domazet *.

Email(s): sofija.baloska@mir.uist.edu.mk, daniela.meckaroska@uist.edu.mk, ervin.domazet@ibu.edu.mk

† Department of AITMIR, University for Information Science & Technology St. Paul the Apostle, Ohrid, 6000, R.N. Macedonia

‡ Department of ISVMA, University for Information Science & Technology St. Paul the Apostle, Ohrid, 6000, R.N. Macedonia

* Faculty of Engineering, International Balkan University, Skopje, 1000, R.N. Macedonia

Abstract

The integration of AI-driven chatbots in e-commerce has transformed the way businesses connect with customers and efficiently manage their activities. These advances improve customer service interactions, promote transactions, and offer personalized suggestions - all of which contribute to increased customer satisfaction and increased sales conversions.

In this study on e-commerce chatbots, we explore types of chatbots, such as rule-based and AI-based models. Discuss their advantages and disadvantages. We also explore the potential for chatbots to improve customer service quality reduce costs and strengthen customer relationships. In addition, we examine real-world examples of e-commerce companies that have improved their efficiency by integrating chatbots.

The future work will explore various areas, including improving Natural Language Understanding (NLU) that enhances the interaction between chatbots and humans through Artificial Intelligence (AI) technologies of emotional intelligence and voice-command transactions; incorporating blockchain for enhanced security measures; and developing our AI chatbot for e-commerce purposes.

In today's changing business landscape, having customer-friendly chatbot systems can provide companies with a competitive advantage as chatbot technology continues to evolve and advance.

Keywords

Artificial intelligence, AI-driven chatbots, Natural Language Understanding, e-commerce, Machine Learning

1 Introduction

The fast progress of AI and Machine Learning (ML) has greatly influenced the e-commerce sector by changing how companies engage with their customers. A groundbreaking technology that has made an impact is chatbots. These are AI-driven assistants created to improve communication, between businesses and customers. These smart systems play a role, in automating customer support services and improving user satisfaction while boosting sales through replies and tailored suggestions at any time of the day.

In the world of online shopping systems, like e-commerce platforms have two types of chatbots; rule-based and AI-driven chatbots. To efficiently answer queries or basic inquiries, rule-based chatbots are designed using scripts and decision paths. AI-powered chatbots, on the other hand, interpret user intent from discussions and offer customized responses using technologies like ML and NLU. As a result, they are more sophisticated and skilled in answering queries.

Chatbots are becoming more and more popular in retail because of how well they streamline processes, reduce costs, and increase customer satisfaction.

The Gartner [1] report predicts that chatbots will be a main customer service channel, and more than 25% of

organizations will use them as their sole customer communication channel, dispensing with the need for human customer service agents. This projection shows the continuing trend of using efficient, scalable AI-powered chatbots for customer service. Additionally, in its report, Juniper Research [2] estimates that organizations can possibly reduce operational expenses by as much as \$11 billion each year by 2025 through the implementation of chatbots to manage customer service functions. The report argues that chatbots are very effective in cutting down on operational costs by virtue of automated customer interactions, thus freeing up the human agents' time and reducing the need for more of them. This potential for cost savings is a clear indication of the economic advantages of incorporating AI-based chatbots in customer support.

The integration of chatbots in online shopping has brought about advantages such as 24/7 customer care services and lower operational costs while dramatically increasing the levels of consumer interaction. Moreover, by performing routine tasks and providing instant responses, chatbots enhance customer satisfaction and allow human agents to work on more complex issues. Chatbots that are integrated with Customer Relationship Management (CRM) systems, payment processors, and stock management systems provide a seamless customer experience from the product discovery to the order completion stage. Prominent online retailers such as Sephora [3] and H&M [4] have leveraged chatbots to engage customers and grow their businesses. While chatbots offer benefits they also encounter obstacles in understanding language effectively as well as issues related to security and gaining user trust. The ongoing development in technology is directing attention towards AI systems and voice commerce along with hyper-personalization features, which allow chatbots to provide interactions that resemble human conversations, resulting in more immersive experiences.

This paper explores the different types of chatbots used in e-commerce, their benefits, their integration with existing systems, and real-world case studies demonstrating their effectiveness. Furthermore, it highlights the future trends that will shape the next generation of chatbot technology in e-commerce, including advancements in voice commerce, sentiment analysis, and AI-driven personalization.

2 Chatbots in E-Commerce

Chatbots have changed the way of interaction between a business and its clients on e-commerce platforms and thus have revolutionized customer service. They have contributed to increasing the conversion rates and keeping the customers happy through better user experiences, faster order processing, and more customer engagement. In the third generation of chatbots, with the development of AI, ML, and NLP, chatbots can actually understand the intent of humans, learn from the conversations, and offer contextually relevant and adaptive responses.

2.1 Introduction of Chatbots

Chatbots are a type of virtual personal assistant that uses AI to mimic human conversation in a chat-like format via text or voice. Chatbots improve customer engagement on e-commerce sites through product suggestions, instant support, and a simpler buying process. There are two primary types of chatbots: The first one is the AI-based chatbots that leverage machine learning and NLP to interpret the input of the user and respond accordingly and the other one is the decision-based chatbots that function based on if-then or set guidelines. It is essential for responding to frequently asked questions, enhancing the overall user experience, and increasing sales conversion rates as more companies move their customer service operations to digital platforms.

The application of chatbots in e-commerce is possible due to the development of AI and NLP such that they can easily understand and answer what the customer is asking. In 2024, the global chatbot industry was estimated to be worth \$7.76 billion. From 2023 to 2030, it is projected to expand at a compound annual growth rate (CAGR) of 23.3% [5]. The necessity for businesses to reduce expenses and the growing need for individualized shopping experiences are the causes of this increase.

Chatbots are now a critical part of the modern e-commerce strategy because they can offer 24/7 customer care, and help with order status and product recommendations. As per a Statista survey, 80% of consumers interacted with a chatbot in 2022 which indicates that automated customer support is becoming more and more popular [6].

2.2 Types of Chatbots

Chatbots can be broadly classified into two main categories: rule-based chatbots and AI-powered chatbots. Each type serves different business needs, offering varying levels of intelligence and interactivity.

2.2.1 Rule-Based Chatbots

Rule-based chatbots work on the given scripts and decision trees and are therefore straightforward, easy to predict, and relatively cheap for the tasks that imply the resort to frequent and rather basic inquiries. They have a set question-and-answer format, which depends on certain keywords or the input of the user. Because of that, they are very suitable for use in cases where the number of questions is limited, such as student inquiries, customer support frequently asked questions (FAQs), or appointment scheduling. For example, in the educational environment, rule-based chatbots excel at responding to the most common questions related to admission, course timings and schedules, and examinations [7].

Because rule-based chatbots operate using straightforward logic and pre-established rules rather than requiring AI technology or large data sets, one of their main advantages is their ease of setup and management. They are therefore reasonably priced and resource-friendly. Additionally, their predictable behavior results in answers, which is a characteristic in settings like schools and hospitals that demand accuracy and conformity. Rule-based chatbots function best in environments with established communication patterns; this allows companies to automatically respond to inquiries without requiring sophisticated AI systems [8].

Some drawbacks of rule-based chatbots are; that they are unable to respond properly to unexpected questions and cannot learn from the interactions that occur between the user and the chatbot itself; a defect that limits its ability to provide unique or context-aware responses in environments where the customer inquiries are volatile. Furthermore, these chatbots require updates to remain effective with changes in the protocols and customer requirements. This issue was highlighted in research conducted on rule-based systems in the e-commerce sector where chatbots failed to converse with customers and understand their context [9].

With these constraints in place, rule-based chatbots are typically used because they are easy to develop, reliable, and inexpensive when you want to define the expected context in the conversation. It is quite useful in schools, healthcare facilities, and customer support divisions where questions are formulaic and require formulaic answers. In its current state, rule-based systems are expected to integrate with AI-driven models to work around their limitations while keeping the cost low.

2.2.2 AI-Powered Chatbots

The development of intelligent, adaptive, and personalized conversations is possible with the aid of chatbots that are powered by AI such as ML, NLP, and Deep Learning (DL) technologies. They can understand the user intent, they can learn from previous interactions, and can generate dynamic context-aware responses. Thus, they are quite capable of performing complicated customer service tasks, personalized recommendations, and flawless conversational experiences than the basic, rule-based chatbots that work off of pre-defined scripts.

A major plus of AI-powered chatbots is their ability to cling to the conversation, understand the intent of the user, and manage multi-turn conversations. They can easily understand and answer complex questions, and understand figures of speech, sarcasm, and other aspects that render the conversation more human-like. Chatbots developed with the aid of artificial intelligence enhance the user's engagement and satisfaction with the platform by providing relevant responses and product suggestions based on the user's preferences.

The next advantage of AI-powered chatbots is that they can provide personalized experiences for customers with the help of big data analysis of the consumers' activities, purchases, and preferences. This enables them to present suggested products and services, notification messages, and support service recommendations that are most likely to result in a purchase and repeat business from the customer. Chatbots developed with the help of AI enhance user experiences by forecasting the needs of the customer and offering support that is relevant to the customer's preferences and purchase history [10].

Apart from the aspect of personalization, AI-powered chatbots also improve the efficiency of operations by answering routine customer inquiries and providing support 24/7. They reduce operational costs by making it possible to do away with a large customer service team while offering high-quality and consistent service. Chatbots also can scale up operations, which means that businesses can interact with numerous customers at the same time without compromising on the quality-of-service delivery [8]. An AI-based chatbot is a cost-effective solution for businesses that are looking to implement customer service automation and enhance operational productivity. However, the use of AI in chatbots also comes with certain risks and limitations. One major limitation is that they require a large amount of high-quality training data to be able to comprehend the intent of a user and improve their accuracy over time. Additionally, the issue of data security and privacy remains a critical issue as chatbots involve the disclosure of customer's data [11]. However, AI-powered chatbots are still developing and improving day by day, and the recent developments include emotionally intelligent AI, voice commerce, and multimodal interactions. Future studies will be directed at improving NLP, Emotional Analysis, and Adaptive Learning to enhance the effectiveness of AI-based chatbots in digital customer interaction.

3 Benefits of Chatbots in E-Commerce

By automating customer service, reducing costs, and improving the user experience, chatbots have revolutionized the e-commerce sector. Because chatbots allow businesses to increase customer interaction while reducing operating expenses, they are an essential part of modern e-commerce platforms.

3.1 24/7 Customer Support

A major plus of chatbots is that they can provide customer support 24/7. Chatbots are instant response agents that are location and time zone compliant and holidays exempt. It solves policies and problems with order tracking and payment. This guarantees the availability which in turn enhances the customers' satisfaction and the trust in the brand. [12]. Besides, it not only improves the user experience but also decreases customers' frustration because they no longer have to wait for a free human agent. Moreover, chatbots are also more effective as they can handle multiple conversations at once, which is inadmissible for human agents [13]. AI-powered chatbots make customers feel recognized and valued by using machine learning algorithms to detect customer sentiment and provide personalized responses. Companies that use 24/7 chat services report higher conversion rates and customer retention rates as a result of increased user engagement.

3.2 Reducing Operational Costs

Chatbots enable businesses to keep and utilize their large investment in human support teams for those customer care needs that are most difficult to handle, while basic queries are addressed by AI-powered chatbots, thus cutting down on customer service costs by 30% with the involvement of human agents in personalized conversations. Thus, companies can guarantee high-level customer support without employing human support, which saves the company money. The advanced optimization of chatbots brings about cost savings by decreasing the need for phone support, emails, and live chats [12]. Besides, chatbots enhance productivity by providing instant responses to order status, product suggestions, and FAQs, thereby minimizing bottlenecks in customer service departments.

3.3 Improving User Experience

Chatbots play an important role in improving the customer experience in e-commerce. AI-powered chatbots use recommendation techniques to assist customers with their purchases.

Chatbots offer recommendations based on previous searches and purchases made by customers. This level of personalized interactions leads to greater customer loyalty. Furthermore, chatbots improve the user experience by streamlining the checkout process, and when integrated with payment gateways, customers can instantly purchase in the chat interface, leading to improved conversion rates and reduced cart abandonment [14].

Another way that AI-powered chatbots improve the user experience and is essential is that they offer multi-language support, making them more globally recognized and accessible and catering to a diverse audience, which is an essential link in e-commerce.

4 Chatbot Integration with Other E-Commerce Systems

Chatbots are now fully incorporated into e-commerce systems and are used in almost all the processes from customer service to the payment and inventory management. They are not just there to respond to the customer queries anymore. With the help of such chatbot integrated systems, companies can increase productivity, cut down on expenses and reduce complexity thus resulting in increased customer happiness.

4.1 Integration with CRM and Customer Support

Through the ability of a chatbot to centralize customer data, interaction history, and preferences the quality of customer service is enhanced by integration with CRM systems. This connectivity enables chatbots to provide tailored responses based on the previous conversation and client profile, thus increasing client engagement and delight. If chatbots are integrated with CRM databases, they can provide accurate information regarding order status, account balance, and product recommendations to the clients. Moreover, they can effectively prioritize and assign tickets to the right human agents to quickly manage the tickets and reduce the response time, and increase the efficiency of the support. This seamless connection enables businesses to maintain consistent and personalized communication with the customers across all the touchpoints and, as a result, improves customer satisfaction and makes customer service management more effective [15].

4.2 Integration with Payment and Checkout Systems

For e-commerce businesses, e-payments remain challenging. Approximately 68% of shoppers abandon their carts, according to certain studies [16]. A significant portion of internet customers abandon their carts before completing their purchases, despite having taken the time to explore for products. Fortunately, new technologies are coming out that can help you make the payment process better. Chatbots are computer programs that carry on discussions with users of messaging apps or website visitors. With chatbots, you can expedite the payment process and even remind customers to finish their purchases.

There are several chatbots designed to integrate payment and checkout systems to ensure the security of transactions within the chat interface. The KAI by Mastercard [17] and the PayPal Chatbot [18] help users manage their payments and view balances securely. Ping from Shopify lets you shop and make payments with Shopify Payments [19], PayPal, and Stripe [20], while WeChat Pay [21] allows users to make direct contactless payments within the chat. In addition, Facebook Messenger chatbots stripped support in-chat purchases. These chatbots enable users to complete transactions instantly within the chat interface, thus providing a smooth and convenient purchasing experience that can reduce cart abandonment rates and increase user happiness. They offer sophisticated authentication and encryption techniques for safe payment processing to protect customer information and prevent fraud. These chatbots can provide immediate payment-related questions and answers through 24/7 customer service and enhance customer trust. Also, they enhance the checkout process to reduce the time it takes to place orders and increase sales and conversion rates. In addition, these chatbots are integrated with multiple channels, including social media, mobile applications, and websites to ensure that the user has a single and coherent experience.

4.3 Integration with Inventory and Order Management

Using chatbots with inventory and order management systems enhances efficiency and customer satisfaction in e-commerce. It means that chatbots can give real-time inventory information, and the customer can know the availability of the product, stock, and delivery time through the chat interface. This reduces cases of out-of-stock and also reduces the number of times that the customer has to inquire about the product's availability. Furthermore, chatbots help in order tracking since they can provide real-time shipment status information and estimated delivery time to the customers, thus increasing the level of transparency and satisfaction to the customers. By automating inventory management tasks, chatbots assist in guaranteeing proper stocking, avoiding overstocking and stockout scenarios, and expediting the order fulfillment process. They also expedite the return and exchange process to make the post-purchase process more convenient for the customer. It is possible to integrate stock data from many sales channels and avoid overbuying by connecting chatbots and inventory systems [22]. Additionally, these chatbots may produce inventory and demand planning data, which can aid in supply chain management and improve decision-making regarding when to restock.

5 Case Studies of Successful E-Commerce Chatbot Implementations

The e-commerce industry has been revolutionized with AI-powered chatbots by enhancing customer engagement, automating services, and increasing sales conversion rates. The overall customer experience can be improved by implementing chatbots that streamline operations and provide personalized recommendations, this has been successful with several leading global retailers. High customer satisfaction rates, reduced service costs, and increased brand loyalty can be achieved when businesses implement chatbots.

5.1 Sephora's AI Chatbot for Personalized Shopping

Sephora's AI-powered chatbot enhances the customer shopping experience by providing personalized beauty recommendations and virtual consultations. It uses AI algorithms to analyze customer preferences, skin type, and past purchases, delivering tailored product suggestions that increase customer engagement and conversion rates.

The chatbot operates 24/7, offering instant responses to customer inquiries, including product availability, order tracking, and beauty tips. This round-the-clock support improves customer satisfaction and reduces wait times. It is also integrated across multiple platforms, including Sephora's website, mobile app, and social media channels, ensuring a consistent omnichannel experience. Sephora's chatbot improves operational efficiency and lessens the workload for human agents by automating repetitive queries and offering tailored recommendations. In addition to increasing sales, this calculated use of AI technology strengthens Sephora's status as a pioneer in digital consumer interaction [3].

5.2 H&M's Virtual Shopping Assistant

H&M's AI-powered chatbot enhances personalized shopping by offering tailored outfit recommendations based on customer preferences and browsing history. It provides 24/7 customer support, ensuring instant responses and real-time order tracking, which improves customer satisfaction and streamlines the shopping process. The chatbot is integrated across multiple platforms, including the website, mobile app, and social media, ensuring a consistent customer experience. By automating routine inquiries and order tracking, H&M reduces operational costs and increases efficiency, allowing human agents to handle more complex issues. This strategic use of chatbots positions H&M as a leader in digital customer engagement [4].

5.3 Walmart's Conversational Commerce Strategy

Walmart has set out on a bold ambition to deploy conversational AI to introduce "retail speak" into every department and shop. A key component of this project is the dedication to guaranteeing openness, equity, and confidence in all AI-powered undertakings. Walmart's Emerging Tech group has been instrumental in this change, promoting the use of conversational AI in various areas of the business. The company's larger vision for a future where technology effortlessly improves the retail experience is emphasized by its confidence in the enormous potential of conversational encounters. Walmart has started several AI-powered initiatives to transform the retail industry. These programs include the unique "Text to Shop" function that makes buying easier for customers, the Converse shopping assistant that helps shoppers, and "Ask Sam," a tool made to help store personnel. By giving teams across the organization the freedom to create their own distinctive conversational experiences, Walmart has internally promoted an innovative culture. This is accomplished by utilizing the power of huge language models as well as developments in natural language processing and interpretation. Additionally, Walmart's AI technology can be quickly deployed on a global basis. Its rapid uptake in nations like Chile, where the platform integrates easily and doesn't require a lot of language-specific programming, is evidence of this [23].

5.4 Amazon's Alexa Shopping Assistant

Data-based programs that are integrated into Internet of Things devices or applications and can interact with users and answer their requests via voice commands are known as digital voice assistants. To watch, retrieve, and process data depending on user inputs, digital voice assistants use state-of-the-art artificial intelligence and speech recognition technology. Compared to earlier iterations of speech-activated technologies, digital voice assistants are intended to be more human-like in that they can learn from feedback to improve their response performance in the future and enable customization [24].

By offering voice-based shopping, Amazon's Alexa chatbot has transformed e-commerce, allowing customers to order and reorder products easily with simple voice commands, improving convenience and accessibility. Eliminating the need for manual tracking updates by providing real-time order tracking, chatbots allow users to check delivery status instantly, reducing customer support inquiries. Higher sales conversions and improved shopping experiences can be achieved with Alexa's AI-powered chatbot, as it delivers personalized product suggestions by analyzing user purchase history and preferences. This also ensures that customers receive tailored recommendations [25].

6 Ethical and Legal Considerations in AI Chatbots for E-Commerce

These days, there are numerous ways to violate human rights due to the widespread shift of retail to the Internet, the replacement of these salespeople by artificial intelligence, the processing of huge databases of potential customers by AI, and the use of chatbots to interact with consumers.

Therefore, a breach of the regulations governing the gathering and processing of personal data, the use of personal information, and interference with privacy can be considered one of the violations of citizens' rights and freedoms in the area of artificial intelligence in e-commerce [26]. Risks to data privacy, algorithmic bias, job displacement, and regulatory compliance are important concerns. To establish a responsible and balanced AI environment, this section examines these issues and possible solutions.

6.1 Data Privacy Risks

To improve service offerings and personalize experiences, AI in e-commerce relies heavily on consumer data. This raises serious questions about how personal data is collected, stored, and used. Implementing strong security measures to protect against data breaches, obtaining informed consent from users for data collection and use, and complying with international data protection laws are essential to ensure data privacy [27]. If not properly managed, some risks can cause data breaches, identity theft, and regulatory noncompliance, which will undermine the trust of consumers in AI-powered systems.

Unintended Sharing of Sensitive Information: This occurs when a user unknowingly shares personal or sensitive data with the AI system. For instance, a user might share their credit card information, believing that the AI is secure. While non-PII (Personally Identifiable Information) AI models like ChatGPT cannot recall or store this information, given it stores non-PII information temporarily to improve performance for 30 days, the data could potentially be intercepted during transmission if the communication channel is not secure [28].

Data Leakage through Model Outputs: LLM models, such as ChatGPT, may occasionally produce outputs that appear to allude to particular data or disclose sensitive information, even though they are unaware of the specifics of the data they were trained on. These outputs, however, are produced using patterns discovered during training and do not represent access to any particular data sources or private databases. Responses from the AI might "hallucinate" certain delicate-looking elements. The model creates things based on the patterns it discovered during training rather than disclosing sensitive real-world data [29].

Adversarial Attacks: In these attacks, malicious actors try to influence or fool the AI into acting in a particular way, usually for negative ends. An adversarial attack might, for example, entail feeding the AI carefully constructed data intended to trick it into producing offensive or dangerous content. Evasion assaults, Trojan horse attacks, and fake review attacks are a few cybersecurity threats that LLM Chatbots may encounter [30].

Although none of the methods listed below is a cure, they can all help safeguard privacy in the context of LLMs. Large language models like ChatGPT pose privacy problems, which are complicated and multidimensional and probably call for a variety of approaches in addition to continuous study and improvement.

6.2 Algorithmic Bias and Fairness

AI bias, sometimes known as algorithm bias or machine learning bias, describes AI systems that generate biased outcomes that mirror and reinforce societal biases held by humans, such as historical and contemporary social inequity. The original training data, algorithm, or algorithm's predictions can all contain bias.

Because biased algorithms in AI chatbots can lead to stereotyping, uneven user experiences, and legal issues, they are a major concern for e-commerce. Bias in chatbots can be observed in training data, algorithmic choices, and user interactions. If there is a lack of representation of gender, ethnicity, or customer behavior in the datasets, then the chatbots may suggest discriminatory recommendations or responses. Furthermore, some machine learning algorithms work on the principle of statistical relationships and thus form assumptions about the users. It is also possible that, over time, chatbots will learn and reinforce the biases in the data collected from the users, just as it has happened in previous instances where chatbots have learned to converse in offensive language and exhibit biased behaviors. The presence of algorithmic bias in chatbots can result in unfair recommendations of products and services and unequal customer service and may lead to legal consequences for violating existing AI laws. As for the risks, businesses should collect different types of data for training, check the fairness of the AI models often, and use tools like AI Fairness 360 to identify bias to mitigate these risks [31].

Moreover, it is also important that chatbots are explanatory in their suggestions and allow consumers to flag biased conversations. Thus, by implementing ethical AI principles and fairness auditing, organizations can develop reliable and unbiased AI chatbots that can provide equal opportunities for customers in the field of e-commerce.

6.3 Job Displacement in Customer Support

The use of e-commerce AI-powered chatbots has increased the automation of many routine customer support tasks that do not need a human agent to handle things like basic inquiries, order tracking or FAQs. Chatbots also enhance efficiency, cost savings, and availability; however, they also displace jobs, especially at entry-level customer service positions. However, AI can't fully replace human agents as many interactions need emotional intelligence and complex problem-solving. While more and more businesses are implementing AI in customer service, the part of a human agent is not disappearing but rather changing. Instead of replacing people, AI is employed to improve customer support staff so that companies can respond to customers faster and more effectively without compromising the personal touch. A human-AI hybrid team is a combination of individuals and AI that can work at the pace of AI, with the critical thinking and empathy of agents [32]. Some examples of AI-powered tools include chatbots, virtual assistants, and automated workflows that handle the minute tasks of answering frequently asked questions, processing simple requests, and even analyzing customers' sentiments.

Companies should focus on areas like AI supervision, chatbot training, and customer experience analysis to reskill their employees. The employment issues should be addressed first when it comes to ethical AI deployment, and the focus should be given to the transition of the workforce and the development of their skills so that AI can improve the service level rather than just take the jobs.

6.4 Regulatory Compliance and Legal Considerations

If AI systems are to be developed and used properly, that is, morally, responsibly, and within the laws, then privacy and data protection have to be observed. This will also assist in increasing the trust of the users and the effectiveness of these technologies in society. Artificial intelligence systems that grasp a large amount of data can learn and produce sensitive information unintentionally. The leakage of personal data can result in adverse effects such as identity theft, financial loss, or damage to one's reputation. Hence, the protection of data and privacy in artificial intelligence systems is a moral issue.

Many regions enforce data privacy regulations, including the California Consumer Privacy Act (CCPA) and the General Data Protection Regulation (GDPR). These laws are quite rigorous regarding how personal data is gathered, managed, and kept. The CCPA grants individuals the ability to understand and remove their personal details and choose not to have their data sold. Similarly, the GDPR safeguards rights like accessing, rectifying, and erasing data. Failing to comply can lead to severe fines and legal repercussions [33].

Although laws governing AI are still being developed, companies must stay vigilant about legislative developments to ensure their chatbot systems adhere to ethical, legal, and consumer protection standards.

Businesses can achieve compliance by adopting robust data security practices, such as end-to-end encryption, protected data repositories, and chatbot audits. They also need to supply comprehensive information about their chatbot operations.

7 Analysis and Results

This section looks at the latest trends and data on AI chatbots in e-commerce. It provides a clear, numerical view of how chatbots are changing the world of online retail. We see the growth of the market, as well as the growing acceptance and preference for conversational AI in customer service. They help improve customer interactions, improve the way businesses operate, and achieve success. The information is based on credible industry reports, surveys, and new studies, is presented with graphs.

7.1. AI-enabled E-Commerce market size

The global chatbot market size was estimated at USD 7.76 billion in 2024 and is expected to witness growth at a CAGR of 23.3% from 2025 to 2030. Thanks to machine learning and AI developments, the chatbot market is expanding significantly. Businesses are increasingly using chatbots to improve customer service and expedite processes. These virtual assistants provide 24/7 availability, reducing response times and improving user satisfaction. Industries such as e-commerce, banking, and data privacy & compliance are particularly profiting from these advancements. The integration of natural language processing has enabled chatbots to handle complex queries with greater accuracy. This has expanded their use beyond customer support to areas like sales and internal operations [5].

In Figure 1, at first, chatbot solutions are the dominant market (the larger black section) as companies are adopting chatbot platforms. Over time, chatbot services (blue) grow, which implies that there is a growing need for chatbot customization, AI training, and integration with e-commerce systems.

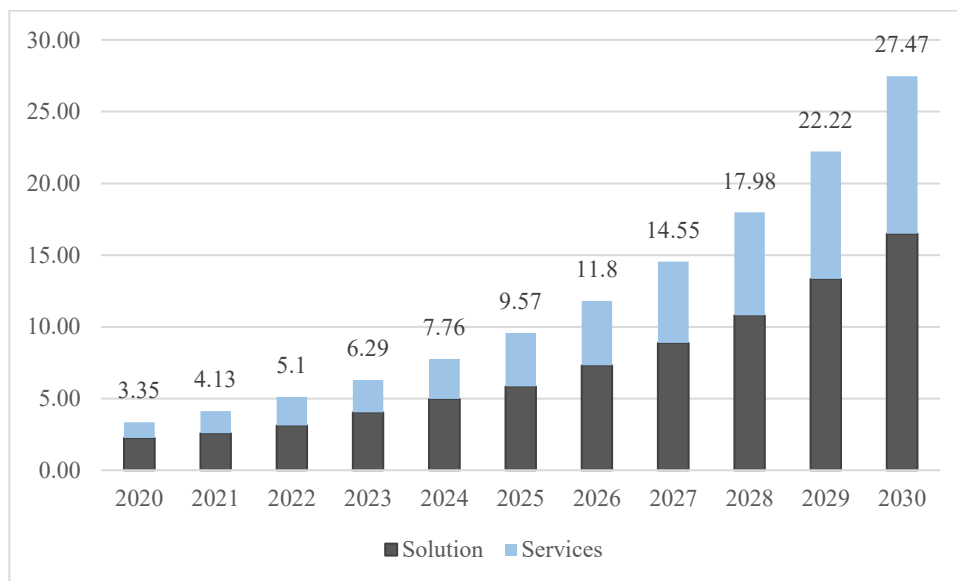


Figure 1: Chatbot Market Size

7.2 Consumer opinions on the use of conversational AI for customer service in 2024

The preference for conversational AI over phone conversations is one of the factors driving AI-powered customer service. 82% of consumers said in 2024 that they would rather utilize a chatbot than wait for a customer service agent to answer the phone. A remarkable 96% of consumers who responded to the survey thought that more businesses ought to use chatbots instead of more conventional customer assistance. These findings, illustrated in Figure 2, highlight the growing consumer trust and preference for AI-driven customer support solutions in the e-commerce sector [34].

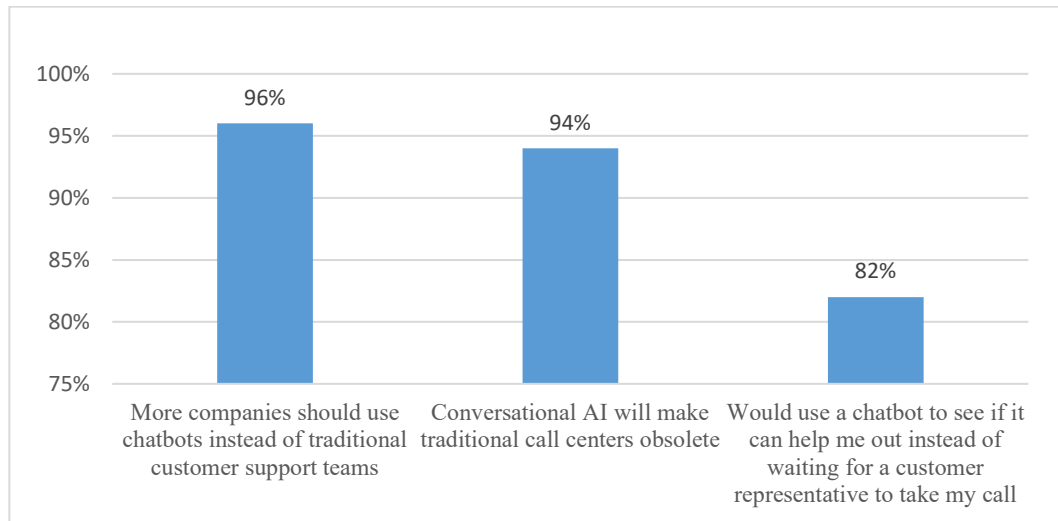


Figure 2: Consumer opinions on the use of conversational AI for customer service in 2024

8 Future Trends in E-Commerce Chatbots

Future e-commerce chatbot research prospects are fascinating since they prioritize areas for enhancing security and user experience while working to increase intelligence and flexibility. Personalization, which uses real-time user data, like behavior patterns and preferences, to provide responses and suggestions during interactions, is a major area of study interest. To do this, contextual AI models that can adapt conversations to the demands of customers must be developed.

Another area of research focuses on enhancing language comprehension and effective conversation management. These days, chatbots struggle to follow discussions that contain sarcasm, many turns, and unclear queries. Ensuring security and protecting data privacy are obstacles for businesses using chatbots. Since chatbots deal with confidential customer details, studies should focus on ways to protect privacy using artificial intelligence methods, ensure secure payment processing, and implement fraud prevention measures. Blockchain technology integration into chatbots is one approach to think about. By strengthening security measures, this integration may boost fraud detection capabilities and transaction transparency. Businesses can use blockchain-based chatbots to keep transaction records in an unchangeable manner and facilitate decentralized authentication procedures, which is crucial for industries handling sensitive financial data.

To use emotionally intelligent AI to identify and react to human emotions, a new avenue is opening up. This development entails giving chatbots the ability to modify their tone to offer responses that are more effective and increase customer satisfaction by integrating sentiment analysis and emotional computing.

Additionally, there is growing interest in chatbot interactions that use a combination of multimedia and voice commands. It is suggested that future studies explore the fusion of virtual reality (VR) chatbots with augmented reality (AR) and voice assistants to enhance shopping and customer service experiences by making them more immersive. As voice-based transactions continue to expand, it will be essential to focus on improving the accuracy of voice recognition and processing technologies to facilitate voice-controlled chatbot interactions.

A major research focus will be on hybrid chatbot-human collaboration. Interestingly, chatbots handle many interactions automatically. Some complex tasks still require human participation. In the future, the focus should be on creating a smooth transition between AI chatbots and human agents to help businesses maintain their efficiency and offer top-quality customer service. In addition to the advancements in chatbot technology for the future, a new AI chatbot application we will be creating for online transactions. This application will prioritize secure, timely payment processing to improve efficiency and prevent fraud. Using automation and security features powered by AI, the chatbot will offer assistance with transactions. Its goal is to improve services by providing seamless and secure transactions that are convenient for users.

9 Conclusion

The application of AI-powered chatbots in e-commerce has changed customer interaction, automation, and business processes and, therefore, become an indispensable part of modern digital commerce. These chatbots improve customer service, facilitate the processing of transactions, and suggest products to the customers, thus enhancing the user experience and increasing the conversion rates. They can work round the clock without breaks, cut down on operational costs, and are compatible with CRM, payment systems, and inventory management, thus playing a crucial role in determining the future of online retail.

The application of AI chatbots in e-commerce is a serious ethical and legal concern. Some of the key concerns include data privacy risks, algorithmic bias, job loss, and regulatory compliance. To achieve responsible use of AI, strict adherence to data protection laws, transparent algorithms, fair labor practices, and compliance with evolving regulations is necessary. Addressing these concerns is essential to establish trust, protect consumer rights, and ensure the ethical growth of AI in the e-commerce sector.

The analysis shows that AI-enabled technologies are growing quickly in e-commerce. The market is increasing steadily, with more investments and users. People like conversational AI; most prefer chatbots over traditional customer support. Trust in AI-driven customer service is rising, and it is becoming essential for the future of e-commerce interactions.

In the future, the development of chatbot technology will be focused on hyper-personalization, natural language understanding, voice commerce, and multimodal communication. Also, the integration of emotionally intelligent AI and fraud prevention mechanisms will make chatbots more clever and secure. Nevertheless, AI-driven automation will continue to be the backbone of real-time, secure, and instant transaction processing, while blockchain will improve security and transparency.

Therefore, AI-powered chatbots are not a convenience but a necessity for any company that wants to offer its customers fast, accurate, and safe service in the virtual space. They will become more vital as technology advances and will keep pushing the growth of e-commerce, making online shopping easier, faster, and more enjoyable for millions of consumers around the globe.

References

- [1] Gartner, 2022. Gartner Predicts Chatbots Will Become a Primary Customer Service Channel Within Five Years. <https://www.gartner.com/en/newsroom/press-releases/2022-07-27-gartner-predicts-chatbots-will-become-a-primary-customer-service-channel-within-five-years>.
- [2] Juniper Research, 2023. Chatbots Research Report: AI & Machine Learning in Customer Service 2023-2025. <https://www.juniperresearch.com/researchstore/fintech-payments/chatbots-research-report>.
- [3] MessageMind, 2023. Case Study: Sephora. <https://messagemind.ai/case-study/sephora/>.
- [4] Redress Compliance, 2023. How H&M Uses AI-Powered Chatbots to Improve Customer Service. <https://redresscompliance.com/how-hm-uses-ai-powered-chatbots-to-improve-customer-service/>
- [5] Grand View Research, 2023. Chatbot Market Size & Growth Report, 2023-2030. <https://www.grandviewresearch.com/industry-analysis/chatbot-market>.
- [6] Statista, 2023. Global Chatbot Usage Statistics. <https://www.statista.com/statistics/1175030/global-chatbot-usage/>.
- [7] Solomon, E., and Tilahun, S. L., 2024. Rule-based Chatbot Design Methods: A Review. *Journal of Computational Science and Data Analytics*, 1(01), 75-84. <https://doi.org/10.69660/jcsda.01012405>.
- [8] Jagdish Singh et al., 2019. Rule-based Chatbot for Student Enquiries. *J. Phys.: Conf. Ser.* 1228 012060. <https://doi.org/10.1088/1742-6596/1228/1/012060>.
- [9] Thorat, S. A., and Jadhav, V., 2020. A Review on Implementation Issues of Rule-based Chatbot Systems. *Social Science Research Network*. <https://api.semanticscholar.org/CorpusID:234976855>.
- [10] Jiang, H., Cheng, Y., Yang, J., and Gao, S., 2022. AI-powered Chatbot Communication with Customers: Dialogic Interactions, Satisfaction, Engagement, and Customer Behavior. *Computers in Human Behavior*, Volume 134, 107329. <https://doi.org/10.1016/j.chb.2022.107329>.

- [11] Le, X.C., 2023. Inducing AI-powered Chatbot Use for Customer Purchase: The Role of Information Value and Innovative Technology. *Journal of Systems and Information Technology*, Vol. 25 No. 2, pp. 219-241. <https://doi.org/10.1108/JSIT-09-2021-0206>.
- [12] Arsturn, 2023. The Cost Benefits of Using AI Chatbots in E-Commerce. <https://www.arsturn.com/blog/the-cost-benefits-of-using-ai-chatbots-in-e-commerce>.
- [13] Brown, W., Wilson, G., and Johnson, O., 2024. Understanding the Role of Chatbots in Enhancing Customer Service. DOI: 10.20944/preprints202408.0321.v1, License CC BY 4.0.
- [14] Singh, G., and Singh, J., 2023. Enhancing Customer Experience and Engagement on E-Commerce Platforms through AI-Integrated Chatbot Solutions, SSRN 4483813.
- [15] Chatbase, 2023. CRM Chatbot. <https://www.chatbase.co/blog/crm-chatbot>.
- [16] Smart Insights, 2023. Why Are Consumers Abandoning Shopping Carts and the Power of Retargeting? <https://www.smartinsights.com/ecommerce/ecommerce-strategy/consumers-abandoning-shopping-carts-power-retargeting/>.
- [17] Chatbot Guide, 2023. Mastercard Bot. <https://www.chatbotguide.org/mastercard-bot>.
- [18] Compose, C., 2023. PayPal Chatbot. <https://www.chatcompose.com/paypal.html>.
- [19] Shopify, 2023. Chatbots in E-Commerce. <https://www.shopify.com/blog/chatbots>.
- [20] Stripe, 2023. Online Payment Processing for Internet Businesses. <https://stripe.com/>.
- [21] Nuvei, 2023. A Guide to WeChat Pay. <https://www.nuvei.com/posts/a-guide-to-wechat-pay>.
- [22] Ochatbot, 2023. Inventory Management for E-Commerce. <https://ochatbot.com/inventory-management-for-ecommerce/>.
- [23] Walmart AI Expert Network, 2023. Case Study: Walmart's Conversational AI Journey. <https://aiexpert.network/case-study-walmarts-conversational-ai-journey/>.
- [24] Aw, E. C.X., Tan, G. W. H., Cham, T. H., Raman, R., and Ooi, K. B., 2022. Alexa, what's on my shopping list? Transforming Customer Experience with Digital Voice Assistants. *Technological Forecasting and Social Change*, Volume 180, 121711. <https://doi.org/10.1016/j.techfore.2022.121711>.
- [25] BotPenguin, 2023. Real-World Examples of AI Enhancing Customer Engagement. <https://botpenguin.com/blogs/real-world-examples-of-ai-enhancing-customer-engagement>.
- [26] Kolodin, D., Telychko, O., Rekun, V., Tkalych, M., and Yamkovyi, V., 2020. Artificial intelligence in E-commerce: Legal aspects. In *III International Scientific Congress Society of Ambient Intelligence 2020 (ISC-SAI 2020)* (pp. 96-102). Atlantis Press.
- [27] Jakkula, A. R., 2020. AI and Ethical Considerations in E-commerce. *Journal of Scientific and Engineering Research*, 7(9), 213-216. ISSN: 2394-2630. CODEN (USA): JSERBR.
- [28] Sebastian, G., 2023. Privacy and Data Protection in ChatGPT and Other AI Chatbots: Strategies for Securing User Information. *International Journal of Security and Privacy in Pervasive Computing*, 15(1), 1-14. <https://doi.org/10.4018/IJSPPC.325475>.
- [29] Alkaissi, H., and McFarlane, S. I., 2023. Artificial hallucinations in ChatGPT: Implications in scientific writing. *Cureus*, 15, 2. doi:10.7759/cureus.35179 PMID:36811129
- [30] Li, J., 2023. ChatGPT as an Attack Tool: Stealthy Textual Backdoor Attack via Blackbox Generative ModelTrigger. arXiv preprint arXiv:2304.14475
- [31] Xue, J., Wang, Y. C., Wei, C., Liu, X., Woo, J., and Kuo, C. C. J., 2024. Bias and fairness in chatbots: An overview. *APSIPA Transactions on Signal and Information Processing*, 13(2).
- [32] Petrescu, M., Krishen, A.S., 2023. Hybrid intelligence: human-AI collaboration in marketing analytics. *J Market Anal* 11, 263–274. <https://doi.org/10.1057/s41270-023-00245-3>
- [33] Sebastian, G., 2023. Privacy and Data Protection in ChatGPT and Other AI Chatbots: Strategies for Securing User Information. SSRN. <https://ssrn.com/abstract=4454761> or <http://dx.doi.org/10.2139/ssrn.4454761>.
- [34] Statista, 2024, Consumer opinions on the use of conversational AI for customer service worldwide as of 2024, <https://www.statista.com/statistics/1538260/consumer-opinions-on-conversational-ai/>.