



# Turning manual web accessibility success criteria into automatic: an LLM-based approach

Juan-Miguel López-Gil<sup>1</sup> · Juanan Pereira<sup>1</sup>

Accepted: 15 February 2024  
© The Author(s) 2024

## Abstract

Web accessibility evaluation is a costly process that usually requires manual intervention. Currently, large language model (LLM) based systems have gained popularity and shown promising capabilities to perform tasks that seemed impossible or required programming knowledge specific to a given area or were supposed to be impossible to be performed automatically. Our research explores whether an LLM-based system would be able to evaluate web accessibility success criteria that require manual evaluation. Three specific success criteria of the Web Content Accessibility Guidelines (WCAG) that currently require manual checks were tested: 1.1.1 Non-text Content, 2.4.4 Link Purpose (In Context), and 3.1.2 Language of Parts. LLM-based scripts were developed to evaluate the test cases. Results were compared against current web accessibility evaluators. While automated accessibility evaluators were unable to reliably test the three WCAG criteria, often missing or only warning about issues, the LLM-based scripts successfully identified accessibility issues the tools missed, achieving overall 87.18% detection across the test cases. Conclusion The results demonstrate LLMs can augment automated accessibility testing to catch issues that pure software testing misses today. Further research should expand evaluation across more test cases and types of content.

**Keywords** Web accessibility · WCAG · Automated testing · Large language models · ChatGPT

## 1 Introduction

Web accessibility is the inclusive practice of removing obstacles that impede persons with disabilities from interacting with or using websites. Making the web accessible has become a top goal as its significance grows across the globe in order to give individuals of all abilities access and opportunity on an equal basis [1]. Numerous studies have emphasized the permanence of accessibility flaws that prevent users from using websites, such as the absence of text equivalents for images, poor color contrast, and audiovisual

content that is not subtitled [2]. According to research, accessible design benefits a broader range of people, including older people with fluctuating capacities [3]. Implementing accessibility promotes independence, prevents prejudice against people with disabilities, and respects social justice principles [4]. Comprehensive best practices are provided by the Web Content Accessibility Guidelines (WCAG) created by the Web Accessibility Initiative of the World Wide Web Consortium for developing accessible digital content [5]. They outline measurable success criteria (subsequently abbreviated as SC) and conformance levels for maximizing accessibility for individuals with disabilities impacting vision, hearing, mobility, cognition, language, and other areas. The recommendations encompass strategies including offering text alternatives, captions and transcripts, keyboard operability, and responsive design to make content perceivable, operable, comprehensible, and resilient.

Web accessibility evaluation entails thoroughly assessing websites and web applications to determine their conformance and overall usability for people with disabilities [5]. This assessment seeks to identify accessibility barriers that may prevent people with vision, hearing, mobility, or

---

Juan-Miguel López-Gil and Juanan Pereira have contributed equally to this work.

---

✉ Juan-Miguel López-Gil  
juanmiguel.lopez@ehu.eus

Juanan Pereira  
juanan.pereira@ehu.eus

<sup>1</sup> Department of Computer Languages and Systems, University of the Basque Country, Manuel Lardizabal 1, Donostia-San Sebastian 20018, Spain

cognitive disabilities from perceiving, comprehending, navigating, and interacting with digital content. While WCAG 2.2 was published in October 2023 [6], conformance testing to the WCAG 2.1 framework remains the most widely used approach, as most evaluation tools have not yet fully implemented updates for the new version's changes. Conformance testing to the WCAG 2.1 includes SC spanning four principles: Perceivability, Operability, Understandability, and Robustness [7]. While some SC can be tested semi-automatically using assistive technologies, others require meticulous manual inspection by trained accessibility specialists [8]. A mixed-methods approach combining automated testing tools, assistive technology user testing, expert code and content audits, usability testing with disabled users, and continuous monitoring is thus required for comprehensive accessibility evaluations [9]. Investigation using multiple evaluation methods allows for the identification of a broader range of WCAG conformance issues and usability barriers for people with disabilities. Recent multi-method evaluations show that accessibility issues are widespread on educational, government, and commercial websites [10]. To achieve truly accessible and inclusive web experiences, sustained efforts are still required.

Generative Artificial Intelligence (GenAI) can be defined as an Artificial Intelligence (AI) system that uses machine learning techniques to create new, realistic media in a particular domain, rather than simply recognizing patterns in data. Generative models learn a distribution over data and can capture the complex structure of the distribution in order to generate new data points. For example, GenAI systems can generate new images, music, text, molecules, or other kinds of media that mimic the statistical properties of real-world data in that domain [11]. Large Language Models (LLMs) are one of the core technologies powering many state-of-the-art generative AI systems today. Systems like ChatGPT,<sup>1</sup> DALL-E,<sup>2</sup> and Stable Diffusion<sup>3</sup> use LLMs to generate high-quality text, images, and other multimedia content. LLMs provide the natural language understanding and generation capabilities to allow generative AI systems to produce meaningful and coherent outputs.

In this paper we investigate whether LLMs can be used to automate WCAG success criteria tests that are so far checked manually. Providing practitioners with effective uses of LLMs in the web accessibility area could enable the automatic evaluation of success criteria that currently require manual evaluation. This would reduce the time and cost burden associated with accessibility evaluation. Considering the aforementioned reasons, the following research question is posed:

**RQ:** Would an LLM-based system be able to evaluate web accessibility success criteria that require manual evaluation?

To address this question, we conducted a controlled study evaluating three specific web accessibility SC using the WCAG accessibility conformance testing rules (WCAG-ACT rules). When evaluating the benchmark test cases based on the HTML-only testing rules defined for the three criteria, most of the common automated testing tools struggled to reliably evaluate the success criteria. They achieved from 0% to 59% when analyzing the accuracy on samples designed to be passed, and 0% accuracy when identifying the intentional failures introduced. However, LLM-based scripts developed for this study were able to identify issues the tools missed or only warned about, achieving an overall 87.18% detection, tests with intentional failures included. This illustrates how LLMs could complement automated testing by detecting web accessibility issues that specialized testing tools may overlook.

The remainder of the paper is structured as follows. Section 2 reviews related work on WCAG success criteria, challenges in manual evaluation, and existing applications of LLMs to web engineering tasks. Section 3 then outlines the method used, including the selection of specific WCAG criteria, materials, and procedure for developing and evaluating the LLM-based scripts, with in-depth description of the procedure for each SC. Section 4 presents results on the performance of the LLM scripts compared to current evaluators. Section 5 discusses the implications of our work, while Sect. 6 describes the limitations of the findings. Finally, Sect. 7 summarizes conclusions and future research directions.

## 2 Related work

In the context of Web the Content Accessibility Guidelines (WCAG), Success Criteria (SC) are specific, testable rules that must be satisfied in order to conform to the guidelines. Each success criterion is associated with one or more guidelines and falls under one of the four overarching principles of WCAG: Perceivable, Operable, Understandable, and Robust (often referred to by the acronym "POUR"). Published by the World Wide Web Consortium (W3C), the guidelines aim to make web content more accessible to people with disabilities [5]. SC in WCAG are defined to be sufficiently specific that they can be objectively evaluated. That is, two different people evaluating the same web content against a success criterion should arrive at the same conclusion. For instance, under the first principle ("Perceivable"), one guideline is "Provide text alternatives for any non-text content". The associated success criteria might include requirements

<sup>1</sup> <https://openai.com/blog/chatgpt>.

<sup>2</sup> <https://openai.com/dall-e-2>.

<sup>3</sup> <https://stability.ai/stablediffusion>.

like “All non-text content that is presented to the user has a text alternative that serves the equivalent purpose”, with further details to guide the interpretation and application of this requirement. Additionally, success criteria in WCAG are assigned to one of three conformance levels: A (lowest), AA (mid range), and AAA (highest), based on the impact they have on accessibility. Meeting more stringent SC at higher levels can improve accessibility for more users or in more situations, but it is often not feasible to achieve 100% AAA conformance for all web content.

Web accessibility evaluation tools are software programs that assess the compliance of websites and web applications with accessibility standards and guidelines. These tools run automated checks on code, markup, and content to identify potential accessibility barriers for people with disabilities [12]. Evaluation tools are commonly used to test WCAG 2.1 conformance, compile auditing reports, and monitor websites for ongoing compliance and quality assurance. The tools use rulesets, heuristics, and machine learning algorithms to automatically detect issues such as missing alternative text, insufficient color contrast, inaccessible PDFs, and other technical flaws that may impede usage for those who rely on assistive technologies [13]. Comparative analyses have found no single ideal web accessibility tool, as each had limitations [14].

The results of different web accessibility evaluation tools can vary due to a range of factors. Some web accessibility tools may focus on specific aspects of web accessibility, such as color contrast or keyboard navigation, whereas others may provide a more comprehensive assessment covering a broader range of accessibility criteria. The tool’s algorithms and heuristics can also influence the results, as different tools interpret and apply the guidelines in different ways [13]. Furthermore, the scope and depth of the analysis can differ between tools, with some performing a more thorough analysis of the website or application and others providing a more cursory evaluation [15]. Another important consideration is the inherent difficulty in automating the evaluation of certain accessibility guidelines, which may be more qualitative and require human judgment [16]. As a result, while tools can provide a valuable baseline, they frequently require manual evaluations to ensure thorough accessibility analysis.

Accessibility barriers encountered by users frequently exceed what technical guidelines and conformance testing can detect. This is due to the subjective nature of perceived accessibility issues, as well as their severity for individual users, which is determined by factors such as how someone navigates, age, abilities, and expertise. While algorithms have been developed to detect and report on accessibility barriers by analyzing real-world user interactions and monitoring visually impaired users, these technological solutions have limitations, as cited in [17]. Many challenges remain difficult to fully resolve or avoid with current capabilities.

More research and development is needed to achieve robust technological accessibility solutions, and user perspectives and community participation are also important [18].

In web accessibility evaluations, false positives are issues that are incorrectly reported as failures or violations when they do not violate standards, whereas false negatives are accessibility barriers that are not identified as problems [15]. These errors are frequently caused by technical limitations in automated testing tools, which struggle to analyze contextual factors, visually perceive pages, or comprehend content semantics [13]. For example, a tool may flag acceptable color contrast ratios as insufficient, or it may fail to recognize when an image lacks a textual equivalent [19]. High false positive rates cause inefficiencies by potentially overloading developers with invalid warnings, whereas false negatives allow barriers to remain unaddressed. Balancing these tradeoffs necessitates the use of tools tailored to the specific needs and workflows of a project, strategic sampling approaches, and human verification procedures to make accessibility an ongoing effort across the organization rather than a one-time compliance checklist [20].

A web accessibility evaluation methodology is the systematic process and procedures used to evaluate websites and web applications for accessibility standards compliance and usability for people with disabilities. Methodologies aim to ensure evaluations are conducted in a consistent, reproducible, and comprehensive manner [21]. Barrier analysis research, user-centric evaluation, and integration with development workflows are all part of accessibility methodologies [22]. This entails using predefined tests, metrics, sampling methods, and reporting frameworks to assess the presence of accessibility barriers that may impede usage for those with vision, hearing, mobility, or cognitive impairments [23]. Key activities also include assistive technology user testing and expert heuristic reviews grounded in WCAG criteria [12]. There is empirical evidence that expecting high levels of agreement for reliable WCAG human testability is not attainable when involving experienced evaluators and neither with novices [24]. Two experienced evaluators would agree on average on slightly more than half of the success criteria. Involving WCAG experts, or different pages, or more specific guidance on how to interpret success criteria, could lead to higher reliability figures [24]. Examples of web accessibility evaluation methodologies include the WCAG-EM, which guides website accessibility evaluations in accordance with WCAG by outlining auditor procedures for scoping, content sampling, auditing, and reporting [25]. It promotes best practices for experienced evaluators across platforms without mandating specific tools and prioritizes the evaluation process over technical compliance. It also supplements WCAG for self-audits and third-party audits. The Unified Web Evaluation Methodology (UWEM) was developed by European experts to evaluate WCAG conformance.

It provided test descriptions, a sampling scheme, reporting options like score cards, and communication instruments, and aimed to become the standard web accessibility evaluation, policy, and certification basis across Europe [26].

## 2.1 LLMs and web engineering

Large Language Models (LLMs) have shown utility in web engineering tasks, though scholarly discourse on the topic appears to be relatively sparse.

A study conducted by the Google Research team [27] explored the ability of LLMs to comprehend HTML. They put the LLMs to the test across three separate tasks, namely: the semantic classification of HTML elements, generation of descriptions for HTML inputs, and autonomous navigation of HTML pages. Remarkably, the research concluded that LLMs pretrained on ordinary natural language datasets adapt well to these HTML-specific tasks.

Tools incorporating generative AI, such as GitHub Copilot and ChatGPT, have been utilized to assist in the development of web applications. Researchers at the Blekinge Institute of Technology, performed a comparative study on the practicality of building entire websites using these AI-powered code-generation tools [28].

OpenAI's ChatGPT and GitHub's Copilot were put to the test, with both evaluated on efficiency, accuracy, maintainability, and ease of use. The researchers found that both tools delivered similar quality code and managed to create websites with minor styling differences, though it was slightly easier to start from scratch with ChatGPT. However, they concluded that current AI code-generation technology is not yet advanced enough to create systems without introducing bugs and potential security risks in a time-efficient manner.

In the scope of web-oriented LLM applications, ChatGPT has proven instrumental in the detection of phishing websites. A novel method for identifying such sites using GPT-4 was proposed by [29]. Recognizing the gap in leveraging LLMs for detecting malicious web content, the researchers used a web crawler to gather information and generate prompts for ChatGPT to analyze. This method allowed for the identification of phishing sites without fine-tuning machine learning models and helped recognize social engineering techniques based on websites and URLs. Recent work on LLM-based systems capable of searching for relevant Graphical User Interface layouts has also demonstrated promising capabilities for engineering user interfaces [30].

In the realm of web accessibility evaluation, LLMs present a significant opportunity yet to be fully explored. Despite the fact that these models have demonstrated remarkable capacities in understanding, generating, and transforming human language, there is a conspicuous lack of research examining their potential application in evaluating web accessibility success criteria. The importance of such

evaluation cannot be overstated, as it allows for enhanced usability and inclusivity for all web users, including those with disabilities. To the authors' knowledge, this represents a significant research gap in the intersection of AI and web accessibility studies. Our aim is to contribute to the filling of this gap, thereby aiding in the development of more inclusive, accessible, and user-friendly digital environments for all web users.

To the authors' knowledge, [31] is the only work so far that suggests using LLMs for improving website accessibility. In their work, they used ChatGPT to automatically remediate accessibility errors found in two web pages. The WAVE web accessibility evaluation tool was used to identify issues on two non-compliant websites, which served as the foundation for ChatGPT-driven remediation. The effectiveness of ChatGPT as an accessibility remediation tool was evaluated by comparing these LLM-generated findings to manual testing.

## 3 Methods

This section details the approach taken to evaluate whether LLMs can automate accessibility testing for specific WCAG success criteria. First, the context and motivation are described, including the need for manual checking of certain criteria. Second, the specific WCAG criteria selected as test cases are presented along with the materials used, including sample HTML code, accessibility evaluators, WCAG-ACT rules, and the LLMs. Finally, the procedure is outlined, explaining, for each specific WCAG criteria, how the LLM-based scripts were developed and evaluated on the test cases in comparison to current automated evaluators. The results are then analyzed in the following section.

### 3.1 Context

For this work we want to prove that there are some WCAG guidelines that up to now, could only be correctly evaluated by a human, and when tried with automatic evaluators, they are marked as passed when, due to problems that we will describe, they should be marked as failed. Our point is that, with the help of LLMs like ChatGPT, Claude or Bard, those guidelines can be correctly and automatically evaluated by the LLMs.

Specifically, as a proof of concept, we are interested in the following WCAG three guidelines, each of a different accessibility principle.

- **WCAG 1.1.1 Non-text Content:** The purpose is to provide text alternatives for non-text content like images, videos, audio clips, etc. so that it can be changed into

**Table 1** Percentage of applicability of overall ACT test cases for each SC

WCAG SC	ACT test cases	% of test cases where only HTML is tested
1.1.1	27	9/27 (33%)
2.4.4	28	20/28 (71%)
3.1.2	10	10/10 (100%)

See <https://gist.github.com/juananpe/17d022dd54f0805090a788c6b56f085e> for details about the applicable SC

other forms people need, such as large print, braille, speech, symbols. This makes content more accessible to blind users or those with low vision. For example, consider the following IMG tag:

```

```

While this code includes the alt attribute with text, SC 1.1.1 emphasizes the text should serve an equivalent purpose to the visual image. In this case, the alt text stating “W3C logo” does properly fulfill the equivalent purpose if and only if the logo.png graphic actually displays the W3C logo. The presence of alt text alone does not guarantee validity. Currently, it is necessary to manually confirm that logo.png displays the W3C logo in order to determine whether the alternative text matches the visual purpose.

- **WCAG 2.4.4 Link Purpose (In Context):** The purpose is to ensure links have discernible text that identifies their purpose or destination. Links should identify their purpose or destination. This helps all users, especially those with cognitive disabilities, understand the link’s purpose.
- **WCAG 3.1.2 Language of Parts:** The purpose is that content in a different language from the main page language is identified through markup, so screen readers can pronounce it properly. This helps readers with cognitive or learning disabilities understand content in multiple languages.

### 3.2 Material

The WCAG accessibility conformance testing rules (WCAG-ACT rules<sup>4</sup>) compile over 1,100 test cases that outline how to reliably evaluate web content against WCAG criteria. Each test case can be marked as “passed”, “failed”, or “inapplicable” to map to accessibility conformance levels.

**Table 2** Amount of selected, modified, and total ACT test cases for each SC

WCAG SC	Selected ACT test cases	Modified ACT test cases	Total ACT test cases
1.1.1	9	9	18
2.4.4	20	20	40
3.1.2	10	10	20

From the full set of WCAG-ACT test cases, we filtered to identify those applicable to our three WCAG SC of interest requiring manual checks—Non-text Content (1.1.1), Link Purpose (2.4.4) and Language of Parts (3.1.2). This produced an initial pool of selected test cases targeting our focus areas.


Table 1 shows the percentage of applicability of overall test cases for each SC. It shows the total number of test cases that tackle each SC, and the percentage of those test cases that only test HTML markup, not other content like JavaScript or multimedia. WCAG guidelines 1.1.1, 2.4.4, and 3.1.2 were addressed in 27, 28, and 10 of the 1132 test cases defined in WCAG-ACT, respectively. As for the percentage of HTML-only test cases, it varied from 33% for 1.1.1, to 71% for 2.4.4, to 100% for 3.1.2.

In addition to selected test cases, the benchmark set also utilizes modified versions that introduce deliberate failures. Comparing results can reveal issues with false positives or negatives. If tools perform the same on both, it likely indicates limitations in accurately identifying problems. Thus, we transformed the selected cases by altering key attributes to intentionally fail the associated success criteria: 1.1.1—Changed alternative text to unrelated filler text; 2.4.4—Changed link text to unrelated filler text; and 3.1.2—Changed language tag to incorrect language. This enabled the analysis of whether tools would properly detect differences between accessible selected cases and modified failures. The specific modifications were:

- **1.1.1:** The text alternatives of each test case were modified to the text: “Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua”, which was not a proper text alternative for any case;
- **2.4.4:** The link descriptions of each test case were modified to the text: “Lorem ipsum dolor sit amet”, which was not a proper description for any case;

<sup>4</sup> W3C WCAG Conformance Test Rules: <https://github.com/w3c/wcag-act-rules/blob/main/content-assets/wcag-act-rules/testcases.json>.

**Fig. 1** HTML code to be tested for WCAG 1.1.1 and screenshot of the rendered HTML. This is the test 32bfac8a98cc212aa7bf9151bf40f665a7f51696 from the WCAG ACT testcases set

HTML code to be tested:	Screenshot of the rendered HTML
<pre> &lt;!DOCTYPE html&gt; &lt;html lang="en"&gt; &lt;head&gt;   &lt;title&gt;Passed Example 1&lt;/title&gt; &lt;/head&gt; &lt;body&gt;   &lt;img alt="W3C logo" src="w3c-logo.png" /&gt; &lt;/body&gt; &lt;/html&gt; </pre>	

- **3.1.2:** The language attribute was changed to “es” in all the test cases, as there was no text in Spanish in any test case.

Both Selected and Modified test cases provide a controlled environment to assess the accuracy and reliability of the web accessibility evaluation tools and libraries for the test cases, as we can discern whether the software is genuinely identifying accessibility issues or simply returning false positives or negatives. Table 2 shows the amount of total test cases we used for each SC, including Selected and Modified ones.

The specific ACT Test IDs for selected test cases are detailed in the Results (Sect. 4).

To test that a web page conforms to the WCAG guidelines we used six different Web Accessibility evaluation tools: A11y,<sup>5</sup> Pa11y,<sup>6</sup> Mauve++,<sup>7</sup> AChecker,<sup>8</sup> AccessMonitor,<sup>9</sup> and Lighthouse.<sup>10</sup>

The results of each web accessibility evaluation tool were labelled according to the recommendations provided by the W3C,<sup>11</sup> further detailed in the W3C Report tool.<sup>12</sup> The result of the evaluation of a WCAG SC can be:

- Passed (P): the corresponding SC was checked and the content was deemed to pass it;
- Failed (F): the corresponding SC was checked and the content was deemed to fail it;
- Cannot Tell (CT): the evaluator cannot provide an outcome for the corresponding SC;
- Not Present (NP): there is no content applicable to the corresponding SC;

- Not Checked (NC): the corresponding SC was not checked by the evaluator.

Finally, we used OpenAI ChatGPT (GPT-3.5 and GPT-4),<sup>13</sup> Anthropic Claude,<sup>14</sup> Google Bard<sup>15</sup> LLMs, and LangChain<sup>16</sup> to build the scripts to analyse the accessibility of the applicable tests.

### 3.3 Procedure

First, the sample of HTML-only WCAG-ACT tests that tackle WCAG 1.1.1, 2.4.4 and 3.1.2 guidelines was evaluated using the six web accessibility evaluators described in the previous subsection. Then, we generated LLM-powered scripts to prompt each LLM in order to evaluate each test. The specificities of the LLM-powered scripts to evaluate the tests corresponding to each WCAG SC are described next.

#### 3.3.1 WCAG 1.1.1 (perceivable)

This rule aims to check that the web page code provides text alternatives for non-text content (like images). We used the WCAG-ACT test case<sup>17</sup> of Fig. 1 as an example.

The page just shows the logo of the W3C consortium with an alternative text in the alt attribute for accessibility. The alt text correctly describes the logo and the evaluators mark this page as valid (‘passed’). But what would happen if the developer wrongly tags the image with this alt value: “a black dog”? The logo and the alternative text won’t match and therefore, the test should be marked as ‘failed’. However, testing it with Mauve++ (or any other validator) will tell us that, as long as there is a text description attached to the logo, it is a valid (accessible) page.

Our approach for this section would be to: 1) given an image (like the W3C logo), try to automatically describe its content

<sup>5</sup> <https://www.a11yproject.com/>.

<sup>6</sup> <https://pa11y.org/>.

<sup>7</sup> <https://mauve.isti.cnr.it/>.

<sup>8</sup> <https://achecks.org/achecker/>.

<sup>9</sup> <https://accessmonitor.acessibilidade.gov.pt/>.

<sup>10</sup> <https://developer.chrome.com/docs/lighthouse/overview/>.

<sup>11</sup> <https://www.w3.org/WAI/test-evaluate/>.

<sup>12</sup> <https://www.w3.org/WAI/eval/report-tool/>.

<sup>13</sup> <https://openai.com/blog/chatgpt/>.

<sup>14</sup> <https://claude.ai/>.

<sup>15</sup> <https://bard.google.com/>.

<sup>16</sup> <https://www.langchain.com/>.

<sup>17</sup> Test case ID: 32bfac8a98cc212aa7bf9151bf40f665a7f51696.

**Fig. 2** HTML code to be tested for WCAG 2.4.4 and screenshot of the rendered HTML. This is the test a8cc66de4d60e-34c7ee0d09fd6ab965ac23d9b4f from the WCAG ACT testcases set

Code:	Screenshot
<pre> &lt;!DOCTYPE html&gt; &lt;html lang="en"&gt; &lt;head&gt;   &lt;title&gt;Passed Example 1&lt;/title&gt; &lt;/head&gt; &lt;body&gt;   &lt;a href="https://www.w3.org/WAI"&gt; Web Accessibility Initiative (WAI) &lt;/a&gt; &lt;/body&gt; &lt;/html&gt;                     </pre>	

using an LLM; 2) now that we have an allegedly correct description of the image, compare it, using another LLM to the description given by the developer in the alt attribute of the HTML; 3) Mark the test as passed or failed based on a threshold.

#### *Details*

For our example about section WCAG 1.1.1 (Perceivable), we want to detect this error (where we changed the PNG image from the test shown in Fig. 1):

```

```

As stated, we will follow this procedure: 1) given the w3c-logo.png image, try to automatically describe its content using an LLM; 2) compare it, using another LLM, to the description given by the developer in the alt attribute (“a black dog”); 3) Mark the test as passed or failed based on a threshold.

#### *LLMs for getting the description of an image*

There are multiple LLMs that can handle the task of getting the description of an image (given the image binary file). We tried the following two open source LLMs, one from Google<sup>18</sup> and the other from Salesforce<sup>19</sup> and implemented a Python script<sup>20</sup> that automates the process. Passing the W3C logo to the Python script, we got the following description: “A sign that says W3C is on a white background.”

#### *LLMs for comparing the similarity of two sentences*

We used the GPT-3.5 LLM to solve the following question:

“Given the following true sentence:

A sign that says W3C is on a white background. Tell me how similar is that sentence to the following one: {sentence}

Write your answer in a JSON object with one key: similarity (a float from 0 to 1)”

where sentence is a placeholder that will be filled with the content of the alt attribute value (in our example “a black dog”). We implemented a LangChain app in Python<sup>21</sup> to solve that question, obtaining the following results:

(“The logo of the W3C”,0.75); (“A black dog”,0); (“The icon of the World Wide Web Consortium”,0.5); (“W3C logo”,0.5); (“a logo”,0.4).

#### *Threshold value for passing the test*

We wondered how to select a threshold value for grading the test as passing or not. To obtain it we resorted to the following method: first, we asked an LLM to generate a set of 100 sentences of different levels of similarity to a fixed one. Then, all the authors of this paper manually evaluated all the sentences, removing the sentences that were not acceptably similar to the fixed one. When in doubt, we discussed the conflicts until a consensus was held. From the remaining sentences, we chose the one with the lower LLM-assigned similarity index. That number marked the acceptable index (0.5) to classify an alt sentence as correct, thus, grading the test as passed.

### 3.3.2 WCAG 2.4.4 (operable)

This rule aims to ensure links have discernible text that identifies their purpose or destination. We used the WCAG-ACT test case<sup>22</sup> of Fig. 2 as an example.

This page just shows a link (anchor element) to w3.org/WAI, with the corresponding text describing the link. But again, accessibility validators are not able to test if the text linked to the anchor is really describing the page the user will see when clicking on the link. So what will happen if instead of using the string “Web Accessibility Initiative (WAI)” for describing the link we change it like this? (we changed the description text from the test shown in Fig. 2):

<sup>18</sup> google/pix2struct-textcaps-base.

<sup>19</sup> salesforce/blip-image-captioning-large.

<sup>20</sup> <https://gist.github.com/juananpe/98130f0b8f67edbd03f19f498dc10891>.

<sup>21</sup> <https://gist.github.com/juananpe/34bb3c04b0afb50e1e7b702e5a8c1c5f>.

<sup>22</sup> WCAG-ACT test case ID: a8cc66de4d60e34c7ee0d09fd6ab965ac23d9b4f.

**Fig. 3** HTML code to be tested for WCAG 3.1.2 and screenshot of the rendered HTML. This is the test ec40c0a032b11cab-c03d71b6884ab9b85ee160ad from the WCAG ACT testcases set

Code:	Screenshot
<pre> &lt;!DOCTYPE html&gt; &lt;html lang="en"&gt;   &lt;head&gt;     &lt;title&gt;Dutch idioms&lt;/title&gt;   &lt;/head&gt;   &lt;body&gt;     &lt;p&gt; The Dutch phrase &lt;span lang="nl"&gt;"Hij ging met de kippen op stok"&lt;/span&gt; literally translates into "He went to roost with the chickens", but it means that he went to bed early. &lt;/p&gt;   &lt;/body&gt; &lt;/html&gt;                 </pre>	<p>The Dutch phrase "Hij ging met de kippen op stok" literally translates into "He went to roost with the chickens", but it means that he went to bed early.</p>

```

<a href="https://www.w3.org/WAI">International Federation
of Association Football (FIFA)</a>
    
```

The automatic evaluators will tag the page as valid even if it is evident that the URL and the description don't match. Here we follow this procedure to test that kind of error: 1) browse the actual link and instruct an LLM to summarize in one sentence the content of the web page; 2) extract the text that describes the link; 3) ask an LLM to compare both sentences (the summary and the extracted text), asking for a similarity metric value; 4) Mark the test as passed or failed based on a threshold.

#### Details

**Browse, summarize and compare.** To check if the description added to the link is correct (it is describing the content of the URL) we used Anthropic Claude LLM and a LangChain script.<sup>23</sup> The script compares the text from the link with the summary text of the actual webpage linked, using the following prompt:

"Browse to <https://www.w3.org/WAI> and summarize the content in one sentence. Then, tell me on a scale from 0 to 1 how suitable it would be to add a link to that website using this text for the anchor: 'International Federation of Association Football (FIFA)'. Write your answer in a single line, as a JSON object with the key *suitability*".

Using the same threshold setting technique as in the previous section, we marked the test as passed or failed.

For the example, this is the output obtained after executing the script (the suitability value indicates that the HTML code does not pass the test: {'summary': 'The Web Accessibility Initiative (WAI) develops strategies, standards, and resources to help make the web accessible to people with disabilities.', 'suitability': 0.1})

### 3.3.3 WCAG 3.1.2 (navigable)

This rule aims to check that if the web page code provides content in a different language from the main page language, the different language is correctly identified through markup, so screen readers can pronounce it properly.

We used the WCAG-ACT test case of Fig. 3 as an example.

This page just shows a paragraph that interleaves two languages: English and Dutch. The problem here is that the content of the lang attribute can be arbitrarily changed to anything (from 'nl' to 'es', for instance), while the automatic evaluators will keep marking the page as 'passed'.

In order to solve this problem, we follow this procedure: 1) send an LLM the HTML content and ask it to check for incorrectly tagged language excerpts; 2) Depending on the answer, mark the test as passed or failed.

#### Details

We feeded three LLMs (Claude, GPT-3.5 and GPT-4) with the incorrect HTML code and asked them to check for inconsistencies in the language tags. Only GPT-4 was capable of detecting the errors consistently with the prompt of Fig. 4:

We obtained the following answer in JSON format, correctly identifying the wrong language in the <span> tag:

```

{"html": {"language": "en", "isCorrect": true},
"title": {"language": "en", "isCorrect": true},
"p": {"language": "en", "isCorrect": true},
"span": {"language": "es", "isCorrect": false}}
    
```

<sup>23</sup> <https://gist.github.com/juananpe/efbd91cbf0bf79fd93ae86106ede0d91>.



**Fig. 4** Prompt used with GPT-4 for detecting incorrectly tagged languages (notice that the <span> tag contains text in Dutch, not in Spanish as hinted by the lang attribute)

```
Based on the following HTML code, generate a JSON object representing the language
identification of each text and if that identification is correct or not:

<!DOCTYPE html>
<html lang="en">
  <head>
    <title>Dutch idioms</title>
  </head>
  <body>
    <p> The Dutch phrase <span lang="es">"Hij ging met de kippen op stok"</span>
literally translates into "He went to roost with the chickens", but it means that he went to bed
early. </p>
  </body>
</html>
```

Don't include any further explanation, just the JSON object

**Table 3** Accessibility results for selected WCAG success criterion 1.1.1 ACT test cases

ACT test ID	E	a11y	A	pa	A	mv	A	ac	A	am	A	lh	A	LLM	A
8c29bcb24ac0f448846a2ffdad4c9693d5aef8c6	P	NP	-	NC	-	P	Y	CT	-	P	Y	F	-	P	Y
b413c09531b239e27bcf79cb57302b429ef59fe6	P	NP	-	F	-	F	-	F	-	P	Y	F	-	P	Y
cab9b2d06e5a44e2056ccbdbb7096f55ab42859c	P	NP	-	F	-	F	-	F	-	P	Y	F	-	P	Y
7d97d6b2f3fa16760bf66026691281a8179f3260	P	NP	-	F	-	F	-	F	-	P	Y	F	-	P	Y
32bfac8a98cc212aa7bf9151bf40f665a7f51696	P	P	Y	NC	-	P	Y	CT	-	P	Y	F	-	P	Y
38cc6a87fcc81fcc2248f0cd74ca48396b7aa432	P	P	Y	NC	-	NC	-	NC	-	P	Y	F	-	F*	-
40d83620b0bcbcf0e7380177384f48596823e7a9	P	P	Y	F	-	NC	-	F	-	P	Y	F	-	P	Y
1b172036f8e219ef9b6f591d7f5df26e4ba11327	P	NP	-	NC	-	NC	-	CT	-	P	Y	F	-	F*	-
af4423575333947073fa3729f502ff0a0c6c2fbf	P	P	Y	NC	-	F	-	CT	-	P	Y	F	-	P	Y
Valid expected percentage		44%		0%		22%		0%		100%		0%		78%	

**Table 4** Accessibility results for modified WCAG success criterion 1.1.1 ACT test cases

ACT Test ID	E	a11y	A	pa	A	mv	A	ac	A	am	A	lh	A	LLM	A
8c29bcb24ac0f448846a2ffdad4c9693d5aef8c6	F	NP	-	NC	-	P	-	CT	-	P	-	F	N	F	Y
b413c09531b239e27bcf79cb57302b429ef59fe6	F	NP	-	F	N	F	N	F	N	P	-	F	N	F	Y
cab9b2d06e5a44e2056ccbdbb7096f55ab42859c	F	NP	-	F	N	F	N	F	N	P	-	F	N	F	Y
7d97d6b2f3fa16760bf66026691281a8179f3260	F	NP	-	F	N	F	N	F	N	P	-	F	N	F	Y
32bfac8a98cc212aa7bf9151bf40f665a7f51696	F	P	-	NC	-	P	-	CT	-	P	-	F	N	F	Y
38cc6a87fcc81fcc2248f0cd74ca48396b7aa432	F	P	-	NC	-	NC	-	NC	-	P	-	F	N	F*	N
40d83620b0bcbcf0e7380177384f48596823e7a9	F	P	-	F	N	NC	-	F	N	P	-	F	N	F	Y
1b172036f8e219ef9b6f591d7f5df26e4ba11327	F	NP	-	NC	-	NC	-	CT	-	P	-	F	N	F*	N
af4423575333947073fa3729f502ff0a0c6c2fbf	F	P	-	NC	-	F	N	CT	-	P	-	F	N	F	Y
Valid expected percentage		0%		44%		44%		44%		0%		100%		78%	
Accurate expected percentage		0%		0%		0%		0%		0%		0%		78%	

**Table 5** Accessibility results for selected WCAG success criterion 2.4.4 ACT test cases

ACT Test ID	E	a11y	A	pa	A	mv	A	ac	A	am	A	lh	A	LLM	A
a8cc66de4d60e34c7ee0d09fd6ab965ac23d9b4f	P	CT	-	NC	-	P	Y	CT	-	NC	-	F	-	P	Y
ada7438401aba500eb03f678b05b9821a758336a	P	CT	-	NC	-	NC	-	NC	-	NC	-	NC	-	F*	-
d13a75a2a0b539a39063eb946505e3d3dd5aeef1	P	F	-	NC	-	F	-	F	-	NC	-	NC	-	P	Y
4493c4b542c8e059e8423c77945ce5895428ab88	P	CT	-	NC	-	F	-	CT	-	NC	-	F	-	P	Y
d6a239059266b317de6a6e73dbf443c5ca8a6f5f	P	F	-	NC	-	F	-	F	-	NC	-	F	-	P	Y
5d16da98a4089b29ff76c611036c65e1c504c7bc	P	CT	-	NC	-	F	-	CT	-	NC	-	F	-	P	Y
e277de30edb9e550d8f9d5a72e1e3adde961d01d	P	F	-	NC	-	F	-	F	-	NC	-	P	Y	F*	-
dee6c55162904cfb77c7f65614c4e6ae2baacea2	P	CT	-	NC	-	P	Y	CT	-	NC	-	F	-	P	Y
b9a3949e2a7521698472a966c78243c4d9ce6fb	P	CT	-	NC	-	P	Y	NC	-	P	Y	NC	-	F*	-
d36abfa44924a4d4088bada05f439ae392dfd662	P	CT	-	NC	-	P	Y	CT	-	NC	-	F	-	P	Y
b130285915a8ca42926a11553a5791f44b65d487	F	CT	-	NC	-	P	-	CT	-	NC	-	NC	-	F	Y
a1e9ff296f0728e180aeb920beacb26bf88ddb12	P	CT	-	NC	-	P	Y	CT	-	NC	-	NC	-	P	Y
e4f70ef2843c6239d0bebe46b97a682bd901e749	P	CT	-	NC	-	P	Y	CT	-	NC	-	NC	-	P	Y
4e89fcc7903980482fe12350f864ca75963d6efd	P	CT	-	NC	-	P	Y	CT	-	NC	-	NC	-	P	Y
c6927fede2d5da439b2d346f39d2ec8980212b31	P	CT	-	NC	-	P	Y	CT	-	P	Y	NC	-	P	Y
e0d32d9583b2b545ca76295cff78e016a44854b6	P	CT	-	NC	-	P	Y	CT	-	NC	-	NC	-	P	Y
91abed1247fb6c9314457a6738343493056fe3bb	P	CT	-	NC	-	P	Y	CT	-	NC	-	NC	-	P	Y
8e6c190e0d2ba8f37707910bd1b984b6885ab548	P	CT	-	NC	-	P	Y	CT	-	NC	-	NC	-	P	Y
b55973d2f813b2fa7d0841202c13f65e41ca8823	P	CT	-	NC	-	P	Y	CT	-	NC	-	NC	-	P	Y
19d5c2888e4434b3e0fb2d9ea5818808e8380422	P	CT	-	NC	-	P	Y	CT	-	NC	-	NC	-	P	Y
Valid expected percentage		0%		0%		65%		0%		10%		5%		85%	

**Table 6** Accessibility results for modified WCAG success criterion 2.4.4 ACT test cases

ACT Test ID	E	a11y	A	pa	A	mv	A	ac	A	am	A	lh	A	LLM	A
a8cc66de4d60e34c7ee0d09fd6ab965ac23d9b4f	F	CT	-	NC	-	P	-	CT	-	NC	-	F	N	F	Y
ada7438401aba500eb03f678b05b9821a758336a	F	CT	-	NC	-	NC	-	NC	-	NC	-	NC	-	F*	N
d13a75a2a0b539a39063eb946505e3d3dd5aeef1	F	F	N	NC	-	F	N	F	N	NC	-	NC	-	F	Y
4493c4b542c8e059e8423c77945ce5895428ab88	F	CT	-	NC	-	F	N	CT	-	NC	-	F	N	F	Y
d6a239059266b317de6a6e73dbf443c5ca8a6f5f	F	F	N	NC	-	F	N	F	N	NC	-	F	N	F	Y
5d16da98a4089b29ff76c611036c65e1c504c7bc	F	CT	-	NC	-	F	N	CT	-	NC	-	F	N	F	Y
e277de30edb9e550d8f9d5a72e1e3adde961d01d	F	F	N	NC	-	F	N	F	N	NC	-	P	-	F*	N
dee6c55162904cfb77c7f65614c4e6ae2baacea2	F	CT	-	NC	-	P	-	CT	-	NC	-	F	N	F	Y
b9a3949e2a7521698472a966c78243c4d9ce6fb	F	CT	-	NC	-	P	-	NC	-	P	-	NC	-	F*	N
d36abfa44924a4d4088bada05f439ae392dfd662	F	CT	-	NC	-	P	-	CT	-	NC	-	F	N	F	Y
b130285915a8ca42926a11553a5791f44b65d487	P	CT	-	NC	-	P	-	CT	-	NC	-	NC	-	P	Y
a1e9ff296f0728e180aeb920beacb26bf88ddb12	F	CT	-	NC	-	P	-	CT	-	NC	-	NC	-	F	Y
e4f70ef2843c6239d0bebe46b97a682bd901e749	F	CT	-	NC	-	P	-	CT	-	NC	-	NC	-	F	Y
4e89fcc7903980482fe12350f864ca75963d6efd	F	CT	-	NC	-	P	-	CT	-	NC	-	NC	-	F	Y
c6927fede2d5da439b2d346f39d2ec8980212b31	F	CT	-	NC	-	P	-	CT	-	P	-	NC	-	F	Y
e0d32d9583b2b545ca76295cff78e016a44854b6	F	CT	-	NC	-	P	-	CT	-	NC	-	NC	-	F	Y
91abed1247fb6c9314457a6738343493056fe3bb	F	CT	-	NC	-	P	-	CT	-	NC	-	NC	-	F	Y
8e6c190e0d2ba8f37707910bd1b984b6885ab548	F	CT	-	NC	-	P	-	CT	-	NC	-	NC	-	F	Y
b55973d2f813b2fa7d0841202c13f65e41ca8823	F	CT	-	NC	-	P	-	CT	-	NC	-	NC	-	F	Y
19d5c2888e4434b3e0fb2d9ea5818808e8380422	F	CT	-	NC	-	P	-	CT	-	NC	-	NC	-	F	Y
Valid expected percentage		15%		0%		30%		15%		0%		30%		85%	
Accurate expected percentage		0%		0%		0%		0%		0%		0%		85%	

**Table 7** Accessibility results for selected WCAG success criterion 3.1.2 ACT test cases

ACT Test ID	E	a11y	A	pa	A	mv	A	ac	A	am	A	lh	A	LLM	A
53a78bad6e92791991df42c50d2e763a9f9d772c	P	CT	-	NC	-	P	Y	NC	-	P	Y	P	Y	P	Y
e0fe6824b5571e0552ab2955697c5ff0776abf79	P	CT	-	NC	-	P	Y	NC	-	P	Y	P	Y	P	Y
207782d0d8899521e2b51b5c384f83d7f4516358	P	CT	-	F	-	F	-	NC	-	P	Y	P	Y	P	Y
8376f95166a75a8541217b77ae6a235f1aac6c3d	P	F	-	NC	-	F	-	NC	-	P	Y	P	Y	P	Y
2febb4d398ed0d788f9ac054ff14cfbd68c0c1f1	P	CT	-	NC	-	P	Y	NC	-	P	Y	P	Y	P	Y
ec40c0a032b11cab03d71b6884ab9b85ee160ad	P	CT	-	NC	-	P	Y	CT	-	P	Y	P	Y	P	Y
df9260fddb4d08ca0669bea363828d089b36317b	P	CT	-	NC	-	P	Y	CT	-	P	Y	P	Y	P	Y
5532e66ea71ed1f352f9911e224cbf290c7cc8e6	P	CT	-	NC	-	P	Y	NC	-	P	Y	P	Y	P	Y
53d05e6fdcc63ff61ef1e5ea8454eea318aa038a	P	CT	-	NC	-	P	Y	NC	-	P	Y	P	Y	P	Y
61c507e0aab456cce20538400fc1067be37953a0	P	CT	-	NC	-	P	Y	NC	-	P	Y	P	Y	P	Y
Valid expected percentage		0%		0%		80%		0%		100%		100%		100%	

**Table 8** Accessibility results for modified WCAG success criterion 3.1.2 ACT test cases

ACT Test ID	E	a11y	A	pa	A	mv	A	ac	A	am	A	lh	A	LLM	A
53a78bad6e92791991df42c50d2e763a9f9d772c	F	CT	-	NC	-	P	-	NC	-	P	-	P	-	F	Y
e0fe6824b5571e0552ab2955697c5ff0776abf79	F	CT	-	NC	-	P	-	NC	-	P	-	P	-	F	Y
207782d0d8899521e2b51b5c384f83d7f4516358	F	CT	-	F	N	F	N	NC	-	P	-	P	-	F	Y
8376f95166a75a8541217b77ae6a235f1aac6c3d	F	F	N	NC	-	F	N	NC	-	P	-	P	-	F	Y
2febb4d398ed0d788f9ac054ff14cfbd68c0c1f1	F	CT	-	NC	-	P	-	NC	-	P	-	P	-	F	Y
ec40c0a032b11cab03d71b6884ab9b85ee160ad	F	CT	-	NC	-	P	-	CT	-	P	-	P	-	F	Y
df9260fddb4d08ca0669bea363828d089b36317b	F	CT	-	NC	-	P	-	CT	-	P	-	P	-	F	Y
5532e66ea71ed1f352f9911e224cbf290c7cc8e6	F	CT	-	NC	-	P	-	NC	-	P	-	P	-	F	Y
53d05e6fdcc63ff61ef1e5ea8454eea318aa038a	F	CT	-	NC	-	P	-	NC	-	P	-	P	-	F	Y
61c507e0aab456cce20538400fc1067be37953a0	F	CT	-	NC	-	P	-	NC	-	P	-	P	-	F	Y
Valid expected percentage		10%		10%		20%		0%		0%		0%		100%	
Accurate expected percentage		0%		0%		0%		0%		0%		0%		100%	

## 4 Results

This section describes the results obtained for each of the ACT rules tested, both the selected and the modified ones, using the methods outlined in the previous section.

Tables 3, 4, 5, 6, 7 and 8 share the same structure. Column “ACT Test ID” shows the IDs for the ACT Test cases in the defined sample of selected and modified test cases. Both selected and modified test cases share the same ID, as modified ones were created by modifying the selected ones, as described in 3.2. Column “E” shows the expected result for each test, which are defined for each test case in the ACT-rules. Columns “a11y”, “pa”, “mv”, “ac”, “am”, and “lh”, correspond to the web accessibility evaluators used (A11y, Pa11y, Mauve++, AChecker, AccessMonitor, and Lighthouse, respectively), as described in 3.2. The LLM column depicts the specific LLM used for each SC, as detailed in 3.3: GPT-3.5 for WCAG 1.1.1, Anthropic Claude for WCAG 2.4.4, and GPT-4 for WCAG 3.1.2.

In all seven columns, the results of each web accessibility evaluation tool were labelled according to the categories defined in the WCAG-EM Report Tool by the W3C: Passed (P), Failed (F), Cannot tell (CT), Not Present (NP), and Not Checked (NC).

There are “A” columns to the right of the “a11y”, “pa”, “mv”, “ac”, “am”, “lh”, and “LLM” columns, respectively. The “A” columns indicate whether the result obtained matches the expected outcome without any manual checking or adjustments by the corresponding evaluators. “Y” means it automatically returned the correct expected result for the Selected case, “N” means it automatically matched the expected result for a Modified case but inaccurately produced the same result as the related Selected case, and “-” means it did not automatically produce the expected outcome.

For example, if a tool marks a Modified case as “Passed” when it should note the deliberate failures introduced and be “Failed”, and this aligns with the tool marking the related Selected case as “Passed” when that version is accessible,

then it indicates a False Positive, as the tool is not properly distinguishing between the Selected and Modified variants. In order to better illustrate the issue, we will examine a Selected case comprising an image with corresponding alternative text that conveys equivalent purpose, thereby satisfying SC 1.1.1. Assuming it includes the code

```

```

which appropriately describes the image content, when evaluated, this selected case passes with a result of “Passed”, as expected. However, the modified version changes the alternative text to

```

```

which now deliberately fails 1.1.1 by having unrelated alt text. Despite this inappropriate change that violates 1.1.1, if the assessment tool still marks the modified case as “Passed” and simply matches the selected case result, it suggests limitations in the tool’s ability to distinguish valid accessible examples from accessibility issues. The two rightful outcomes should be the selected case passing with “Passed” and the modified case failing with “Failed” to properly identify the inappropriate alternative text.

The cells in which LLM-based scripts did not work correctly were also tagged with an asterisk (\*).

“Valid Expected Percentage” row indicates the accuracy on samples designed to be passed (Selected cases) and to be failed (Modified cases). On the other hand, “Accurate Expected Percentage” row denotes correctly identifying the intentional failures introduced. Only the tables corresponding to Modified cases have this row. Both percentages are calculated by taking the number of test cases matching the expected outcome, dividing by the total number of test cases in the benchmark, and then multiplying by 100 to get a percentage.

As for the specific SC and test cases described in each table, regarding SC 1.1.1, Table 3 shows the results for Selected test cases, while Table 4 displays the results for the Modified test cases. As for SC 2.4.4, Table 5 exhibits the results for Selected test cases, whereas Table 6 presents the results for Modified test cases. Finally, with respect to SC 3.1.2, Table 7 features the results for Selected test cases and Table 8 for Modified test cases.

## 5 Discussion

The results demonstrate the potential for using large language models (LLMs) to improve automatic evaluation of accessibility guidelines conformance. Specifically, the proof-of-concept focused on ACT rules that cover HTML-only

elements related three WCAG SC (1.1.1, 2.4.4, and 3.1.2) that cannot currently be reliably tested by automated tools alone. Our LLM-powered scripts were 78%, 85%, and 100% successful in determining the expected outcomes for each test for SC 1.1.1, SC 2.4.4, and SC 3.1.2, respectively, despite the fact that properly selecting the appropriate LLM based on the SC was critical for that goal. The overall performance was 87.18%, as 34 out of 39 test cases were correctly passed.

Although our study showed that LLMs could be used to automate testing for three particular WCAG success criteria that needed to be checked by hand, there are more than 75 success criteria that offer thorough recommendations for web accessibility. Our small sample only scratched the surface. However, the criteria we tested had to do with text alternative provision, link context, and language identification. These criteria—especially the first two—represent common shortcomings in web accessibility [32].

For WCAG SC 1.1.1, which requires meaningful alternative text for non-text content, the LLM-based approach was able to detect mismatches between images and their alternative text by generating a description and comparing it to the provided alternative text. Using ChatGPT-3.5 to compute a similarity score, a reasonable threshold could be set to determine if the alternative text sufficiently described the image. Regarding SC 2.4.4, which requires links to have discernible text identifying their purpose, the LLM Claude, assisted by a Langchain script, was able to browse the target webpage, summarize it, and judge whether the link text matched the summary. Again, a threshold could be set to determine if the link text was acceptable. In relation to SC 3.1.2, which requires marking content in a different language, only ChatGPT-4 was able to reliably detect incorrect language tags in sample HTML by inspecting the text.

The tests performed with LLMs have been carried out as a proof of concept. Still, the LLM-based scripts defined in Sect. 3.3 performed well for Selected and Modified test cases alike. As for the 5 test cases that did not pass, they are denoted with an asterisk (\*) in Tables 3, 4, 5, and 6. Even though the proposed procedure did not work straight away, all the test cases could still be passed using manual prompt editing and specialized procedures to detect the HTML elements with potential problems for each specific case:

- **SC 1.1.1:**
  - *38cc6a87fcc81fcc2248f0cd74ca48396b7aa432*: The prompt was modified to specify that the image is obtained using CSS background-image property.
  - *1b172036f8e219ef9b6f591d7f5df26e4ba11327*: The prompt was modified to specify that the image is

obtained using the `aria-labelledby` attribute to associate the `<object>` element with the `<span>` element

the rest of the script remained the same in both cases;

- **SC 2.4.4:**

- *e277de30edb9e550d8f9d5a72e1e3adde961d01d:*  
The prompt was modified to specify that the link description is labelled by ARIA.
- *b9a3949e2a7521698472a966c782434c4d9ce6fb:*  
The prompt was modified to specify the use of an image as a client-side image map.
- *ada7438401aba500eb03f678b05b9821a758336a:*  
We used GPT-4 and OpenInterpreter<sup>24</sup> to decide if the link that we are redirected to when clicking the button is correctly identified by the text that describes it with the following prompt: “browse to this page <https://raw.githubusercontent.com/w3c/wcag-act-rules/59ccb25310874c7387c9e4a17613af8f025167f/content-assets/wcag-act-rules/testcases/c487ae/ada7438401aba500eb03f678b05b9821a758336a.html> and tell me if the text that describes where I will go if I click the button is right” obtaining the following answer: “The URL that the button will navigate to when clicked is <https://www.w3.org/WAI/>. The text on the button is ‘Click me for WAI!’. Therefore, the text that describes where you will go if you click the button is correct.”

- **SC 3.1.2:** There was no need for specific ad-hoc prompts for the test cases belonging to SC 3.1.2.

Our research has been designed to be reproducible. To that end, we have shared the unique identifiers for each test case, as well as the web accessibility evaluators and LLMs used in each scenario. In addition, we documented the prompts included in our scripts that are critical for evaluating web accessibility. This includes detailed records of instances where custom, ad-hoc prompts were required to pass specific accessibility tests. We hope to facilitate replication of our methods and findings by providing this level of transparency, allowing other researchers to build on our work.

The variability in the results produced by web accessibility evaluation tools is due in part to their varying abilities to accurately assess Modified cases. When cases labeled “N” produce an evaluative outcome that is similar to the result of a Selected case, the likelihood of a False Positive or False Negative increases. This parallelism indicates a limitation

in the tool’s ability to evaluate modifications correctly. When web accessibility evaluation tools produce unexpected results, it reveals potential flaws in the tools’ testing methodologies. It emphasizes the importance of these tools distinguishing between pages that were intentionally modified to be accessible (Modified) and real-world pages that were selected as-is (Selected). It is difficult to be confident that the tools’ accessibility assessments will generalize to arbitrary web pages found in the wild unless results from these two types of test cases are separated. When evaluating real-world web accessibility, distinguishing between Modified and Selected cases helps validate that our results have integrity and applicability. Furthermore, the Automatically Passed Percentage, which is calculated similarly to the Passed Percentage but only accounts for test cases that the tool has passed without manual intervention, quantifies the effectiveness of these tools. This metric serves as an indicator of the tool’s ability to identify and approve accessibility test cases autonomously, emphasizing the importance of robust automatic detection to supplement the manual evaluation process.

This analysis is not intended to create a hierarchy of web accessibility evaluation tools or to declare one superior to another. This comparison is purely informative and is limited to a specific set of test cases chosen for this study. Any conclusions reached are only applicable to this sample and should not be extrapolated to the overall performance of the evaluators in question. We recognize that benchmarking can be difficult due to differences in test case design, evaluator implementation, and the dynamic nature of web accessibility standards. As such, rather than serving as a ranking or endorsement, this examination is intended to contribute to a better understanding of how different tools perform under different conditions.

In the light of the results obtained, LLMs exhibit potential to play a critical role in web accessibility evaluation by assisting individuals in interpreting the results of web accessibility evaluation tools and analyzing SC that cannot be evaluated automatically. They can help evaluators by providing a deeper understanding of the complex datasets generated by automated tools, identifying patterns or discrepancies that might not be obvious at first. LLMs can help in formulating and suggesting evaluation strategies for SC that cannot be automatically assessed due to their qualitative nature, or in generating human-like reasoning to approximate the judgment a human expert might offer. This dual utility highlights LLMs’ versatility as an augmentation tool in the domain of web accessibility, improving both the efficiency and thoroughness of evaluations.

Our study expands significantly on the only precedent found in the literature [31], which examined web accessibility compliance based on WCAG 2.1 by examining two web pages using a single LLM, in this case ChatGPT, with

<sup>24</sup> <https://openinterpreter.com/>.

prompts specific to the accessibility error descriptions found by one web accessibility evaluation tool (in this case, Wave). By incorporating four LLMs and employing a diverse set of 39 test cases within a benchmark framework, our work takes a more comprehensive approach. Furthermore, we expanded our analysis to include three distinct SC derived from different accessibility principles defined in WCAG: SC 1.1.1 for Perceivable, SC 2.4.4 for Operable, and SC 3.1.2 for Operable. This broader scope enables a more nuanced understanding of how these advanced LLMs deal with a variety of accessibility challenges, providing a richer comparative insight into their performance and effectiveness.

## 6 Limitations

The results show LLMs can augment automated accessibility testing to catch issues that pure software testing misses today. However, there are some limitations and areas for further research:

- Our test samples have been limited to the WCAG ACT Test suite. We are aware that the techniques need more systematic evaluation on a larger sample of test cases, ideally using real-world pages as our test benchmark. However, our aim was to present, as a proof of concept, that LLMs are currently capable of automating and correctly detecting accessibility errors that were so far checked manually or just ignored;
- Our research focused on a limited number of Success Criteria (SC), providing a glimpse of the potential for LLMs in web accessibility evaluation. The diversity and complexity of web accessibility standards, on the other hand, require a broader investigation. More research is needed to determine whether the insights gained from LLM-based assessments of these specific SC can be generalized to others;
- The optimal choice of the LLMs to use and ideal prompts must be determined for each task. Other LLMs could potentially work but were not all tested here;
- We have focused on HTML content, and have not analysed other types of content or technologies included in the WCAG (such as PDF or dynamic scripting);
- Threshold values for passing or failing a test may require additional tuning and validation to balance precision and recall.

### 6.1 Threats for replicability

While our research has been designed to be reproducible, certain threats to replicability must be recognized, especially in the rapidly changing technological landscape. The

challenge with LLMs is that they are constantly evolving; as these models are updated to new versions, their responses may change, potentially affecting the performance of our scripts. Because of the temporal variability, replicating our experiments may result in different results as the models evolve. Similarly, the web accessibility evaluation tools that we used have their own set of variables. The versions of underlying libraries may change, web evaluation services may experience brief outages, and updates to the services or libraries themselves may result in discrepancies in evaluation results. Issues such as timeouts can also have an impact on the consistency of results. Additionally, the context in which our research takes place is subject to change, particularly in terms of the ACT rules, which are updated on a regular basis to reflect evolving standards and best practices in web accessibility. Our study used the most recent version of the ACT rules as of June 2023. Since then, the ACT rules have changed, totaling 1105 as of November 2023. Because future researchers using the updated ACT rules may encounter a different evaluation landscape than the one we documented, the modifications to the test cases may have an effect on replicability.

## 7 Conclusion and future work

To address the research question of whether an LLM-based system would be able to evaluate web accessibility success criteria that require manual evaluation, we conducted a controlled study evaluating three specific web accessibility success criteria using the WCAG accessibility conformance testing rules (WCAG-ACT). Using the HTML-only testing rules defined for the three criteria, the LLM-based scripts successfully identified accessibility issues that automatic accessibility evaluators missed or labelled as warnings, achieving an overall 87,18% detection across applicable test cases. This demonstrates that for the evaluated success criteria requiring manual checks, the LLM-based approach was able to accurately evaluate compliance in a way that goes beyond current automated testing tools.

This pilot study demonstrates promising capabilities of LLMs for improving automated accessibility testing. With further research and development, integrating LLM-based techniques could significantly increase thoroughness of conformance checking to WCAG and other accessibility guidelines. This will in turn help developers, testers, and organizations build more accessible digital content.

Further research should be invested in trying to replicate the same tests using open source LLMs instead of OpenAI ChatGPT, Google Bard or Anthropic Claude (closed source LLMs). Open source LLMs will not only lessen the expenses of calling non-free APIs but also will help with privacy issues. Both open source and closed source LLMs

should be put to the test for assessing the generalization of the techniques explained in this work across different, real-world web content and contexts.

Future research would also aid in the development of more generalized prompts that will help LLMs resolve a broader range of tests, building on the foundation laid by our current research. The goal is to reduce reliance on ad hoc solutions, which, while effective, are custom tailored to specific situations and lack universal applicability. We can streamline the evaluation process and expand the range of accessibility issues that can be automatically identified and addressed by LLMs by focusing on the creation of versatile, broadly applicable prompts, thus advancing the field of web accessibility evaluation.

**Funding** Open Access funding provided thanks to the CRUE-CSIC agreement with Springer Nature.

**Data availability** Both the analysis scripts and W3C data used in this research are conveniently available via links within the article.

## Declarations

**Conflict of interest** On behalf of all authors, the corresponding author states that there is no conflict of interest.

**Open Access** This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

## References

- Kelly, B., Nevile, L., Sloan, D., Fanou, S., Ellison, R., Herrod, L.: From web accessibility to web adaptability. *Disabil. Rehabil. Assist. Technol.* **4**(4), 212–226 (2009). <https://doi.org/10.1080/17483100902903408>. (PMID: 19565383)
- Wentz, B., Lazar, J.: Are separate interfaces inherently unequal? An evaluation with blind users of the usability of two interfaces for a social networking platform. In: Proceedings of the 2011 iConference (iConference'11), pp. 91–97. Association for Computing Machinery. <https://doi.org/10.1145/1940761.1940774>. Accessed 20 July 2023
- Bennett, C., Mayhorn, B.J., Morrell, R.W.: Older adults online in the internet century. In: *Older Adults, Health Information, and the World Wide Web*, p. 16. Psychology Press
- Jaeger, P.T.: Designing for diversity and designing for disability: new opportunities for libraries to expand their support and advocacy for people with disabilities. *Int. J. Inf. Divers. Inclusion.* **2**(1), 52–66 (2018)
- Kirkpatrick, A., O'Connor, J., Campbell, A., Cooper, M.: Web Content Accessibility Guidelines (WCAG) 2.1. Technical report, World Wide Web Consortium (2023). <https://www.w3.org/TR/2023/REC-WCAG21-20230921/>
- Campbell, A., Adams, C., Montgomery, R.B., Cooper, M., Kirkpatrick, A.: Web Content Accessibility Guidelines (WCAG) 2.2. Technical report, World Wide Web Consortium (2023). <https://www.w3.org/TR/2023/REC-WCAG22-20231005/>
- Campbell, A., Adams, C., Montgomery, R.B., Cooper, M.: Understanding wcag 2.1. Technical report, World Wide Web Consortium (2023). <https://www.w3.org/WAI/WCAG21/Understanding/>
- Manez-Carvajal, C., Fernandez-Piqueras, R., Cervera-Merida, J.F.: Evaluating web accessibility of Spanish universities. *J. Eng. Appl. Sci.* **14**, 4876–4881 (2019)
- Naftali, M., Findlater, L.: Accessibility in context: Understanding the truly mobile experience of smartphone users with motor impairments. In: Proceedings of the 16th International ACM SIGACCESS Conference on Computers & Accessibility (ASSETS'14), pp. 209–216. Association for Computing Machinery, New York, NY, USA (2014). <https://doi.org/10.1145/2661334.2661372>
- Lewthwaite, S., Horton, S., Coverdale, A.: Researching pedagogy in digital accessibility education. *SIGACCESS Access. Comput.* (2023). <https://doi.org/10.1145/3582298.3582300>
- Goodfellow, I., Bengio, Y., Courville, A.: *Deep Learning*. MIT Press (2024)
- Power, C., Freire, A., Petrie, H., Swallow, D.: Guidelines are only half of the story: accessibility problems encountered by blind users on the web. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI'12), pp. 433–442. Association for Computing Machinery, New York, NY, USA (2012). <https://doi.org/10.1145/2207676.2207736>
- Vigo, M., Brown, J., Conway, V.: Benchmarking web accessibility evaluation tools: measuring the harm of sole reliance on automated tests. In: Proceedings of the 10th International Cross-Disciplinary Conference on Web Accessibility (W4A'13). Association for Computing Machinery, New York, NY, USA (2013). <https://doi.org/10.1145/2461121.2461124>
- Kumar, S., Shree, D.V.J., Biswas, P.: Comparing ten WCAG tools for accessibility evaluation of websites. *Technol. Disabil.* **33**(3), 163–185 (2021). <https://doi.org/10.3233/TAD-210329>
- Brajnik, G.: Comparing accessibility evaluation tools: a method for tool effectiveness. *Univ. Access Inf. Soc.* **3**(3), 252–263 (2004). <https://doi.org/10.1007/s10209-004-0105-y>
- Navarrete, R., Lujan-Mora, S.: Evaluating findability of open educational resources from the perspective of users with disabilities: a preliminary approach. In: 2015 Second International Conference on eDemocracy & eGovernment (ICEDEG), pp. 112–119 (2015). <https://doi.org/10.1109/ICEDEG.2015.7114457>
- Vigo, M., Harper, S.: Evaluating accessibility-in-use. In: Proceedings of the 10th International Cross-Disciplinary Conference on Web Accessibility (W4A'13). Association for Computing Machinery, New York, NY, USA (2013). <https://doi.org/10.1145/2461121.2461136>
- Sato, D., Takagi, H., Kobayashi, M., Kawanaka, S., Asakawa, C.: Exploratory analysis of collaborative web accessibility improvement. *ACM Trans. Access. Comput.* **3**(2), 5–1530 (2010). <https://doi.org/10.1145/1857920.1857922>
- Kane, S.K., Shulman, J.A., Shockley, T.J., Ladner, R.E.: A web accessibility report card for top international university web sites. In: Proceedings of the 2007 International Cross-Disciplinary Conference on Web Accessibility (W4A) (W4A'07), pp. 148–156. Association for Computing Machinery, New York, NY, USA (2007). <https://doi.org/10.1145/1243441.1243472>

20. Stephanidis, C., Salvendy, G., Antona, M., Chen, J.Y., Dong, J., Duffy, V.G., Fang, X., Fidopiastis, C., Fragomeni, G., Fu, L.P., et al.: Seven hci grand challenges. *Int. J. Hum.-Comput. Interact.* **35**(14), 1229–1269 (2019)
21. Giorgio Brajnik, Y.Y., Harper, S.: The expertise effect on web accessibility evaluation methods. *Hum-Comput. Interact.* **26**(3), 246–283 (2011). <https://doi.org/10.1080/07370024.2011.601670>
22. Abou-Zahra, S.: In: Harper, S., Yesilada, Y. (Eds.) *Web Accessibility Evaluation*, pp. 79–106. Springer, London (2008). [https://doi.org/10.1007/978-1-84800-050-6\\_7](https://doi.org/10.1007/978-1-84800-050-6_7)
23. Mankoff, J., Fait, H., Tran, T.: Is your web page accessible? a comparative study of methods for assessing web page accessibility for the blind. In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI'05)*, pp. 41–50. Association for Computing Machinery, New York, NY, USA (2005). <https://doi.org/10.1145/1054972.1054979>
24. Brajnik, G., Yesilada, Y., Harper, S.: Is accessibility conformance an elusive property? A study of validity and reliability of WCAG 2.0. *ACM Trans. Access. Comput.* **4**(2), 8–1828 (2012). <https://doi.org/10.1145/2141943.2141946>
25. Velleman, E., Abou-Zahra, S.: *Website Accessibility Conformance Evaluation Methodology (WCAG-EM) 1.0*. Technical report, W3C (2014). <https://www.w3.org/TR/WCAG-EM/>
26. Nietzio, A., Strobbe, C., Velleman, E.: *The Unified Web Evaluation Methodology (UWEM) 1.2 for wcag 1.0*. In: Miesenberger, K., Klaus, J., Zagler, W., Karshmer, A. (Eds.) *Computers Helping People with Special Needs*, pp. 394–401. Springer, Berlin (2008)
27. Gur, I., Nachum, O., Miao, Y., Safdari, M., Huang, A., Chowdhery, A., Narang, S., Fiedel, N., Faust, A.: *Understanding HTML with Large Language Models*. arXiv. <https://doi.org/10.48550/arXiv.2210.03945>. arXiv:2210.03945. Accessed 2023-08-05
28. Fajkovic, E., Rundberg, E.: The Impact of AI-generated Code on Web Development: A Comparative Study of ChatGPT and GitHub Copilot. <https://urn.kb.se/resolve?urn=urn:nbn:se:bth-24801>. Accessed 2023-08-05
29. Koide, T., Fukushi, N., Nakano, H., Chiba, D.: Detecting Phishing Sites Using ChatGPT. arXiv. <https://doi.org/10.48550/arXiv.2306.05816>. arXiv:2306.05816. Accessed 2023-08-05
30. Brie, P., Burny, N., Sluÿters, A., Vanderdonckt, J.: Evaluating a large language model on searching for gui layouts. *Proc. ACM Hum.-Comput. Interact.* (2023). <https://doi.org/10.1145/3593230>
31. Othman, A., Dhouib, A., Nasser Al Jabor, A.: Fostering websites accessibility: a case study on the use of the large language models chatgpt for automatic remediation. In: *Proceedings of the 16th International Conference on Pervasive Technologies Related to Assistive Environments (PETRA'23)*, pp. 707–713. Association for Computing Machinery, New York, NY, USA (2023). <https://doi.org/10.1145/3594806.3596542>
32. Keith, S., Floratos, N., Whitney, G.: Certification or conformance: Making a successful commitment to wcag 2.0. In: *Proceedings of the International Cross-Disciplinary Conference on Web Accessibility (W4A'12)*. Association for Computing Machinery, New York, NY, USA (2012). <https://doi.org/10.1145/2207016.2207029>

**Publisher's Note** Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.