Measure cookie setting behaviour of web pages showing cookie privacy warnings

Daniel Kirkman

4th Year Project Report
Computer Science
School of Informatics
University of Edinburgh

2022

Abstract

Cookie dialogs are an increasingly common sight across the web but it can often be unclear to users what their true purpose is. These dialogs are intended to allow users to select the level of cookie enablement that matches their personal privacy preferences. However, many cookie dialogs employ subtle design techniques to nudge users towards accepting more cookies than they need to. These techniques are known as Dark Patterns.

The process of manually analysing a website to locate Dark Patterns is time-consuming and can often lead to inconsistent results between researchers. This project will design and implement an automated system that will locate cookie dialogs and then analyse them for Dark Patterns.

The resulting system has been tested on over 10,000 websites and has successfully collected over 2,000 cookie dialogs. The system is capable of detecting 10 different types of Dark Patterns automatically and has found over 3,700 instances of Dark Patterns. The automated system allows us to assess the prevalence of Dark Patterns across the web, at a scale not possible with manual analysis.

Research Ethics Approval

This project was planned in accordance with the Informatics Research Ethics policy. It did not involve any aspects that required approval from the Informatics Research Ethics committee.

Declaration

I declare that this thesis was composed by myself, that the work contained herein is my own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or professional qualification except as specified.

(*Daniel Kirkman*)

Acknowledgements

I would first and foremost like to thank my supervisor, Kami Vaniea, for her support and guidance throughout my project. I am also very grateful for our weekly meetings that were essential to keep me on track and making consistent progress throughout the year. Additionally, I would like to thank my friends and family for all their support and encouragement during my very busy final year.

Table of Contents

Chapter 1

Introduction

Cookies have become a crucial part of the modern internet, they enable web pages to remember information about a user. Without cookies, web pages would be unable to retain state. This means that each time a web page is reloaded it will have forgotten everything the user has done previously, causing frustration for users. They allow the implementation of important functions such as logging in to an account or keeping track of an online shopping basket. However, cookies can also be used to collect sensitive personal data and track a user's activity as they navigate across the web. This enables online advertisers to use targeted advertising with the data collected from cookies which has led to privacy concerns.

To address these concerns, legislation has been introduced to protect users' data online. The Privacy and Electronic Communications Regulations (PECR) was brought into law in 2002 which states that user's consent must be obtained before any cookie that is not essential for the website's basic functionality is set. To comply with this legalisation many websites began to add cookie dialogs to their websites [12].

Cookie dialogs (or cookie privacy warnings) are popups on web pages that allow a user to accept or refuse their consent to non-essential cookies. Ideally, cookie dialogs should provide a range of options for users to choose from and there should be no bias towards accepting more cookies. However, this is rarely the case and many cookie dialogs employ subtle design techniques to nudge users towards accepting more cookies, these techniques are known as Dark Patterns [3].

Previous studies by Soe et al. [21] and Fernandez et al. [11] which looked at Dark Patterns on cookie dialogs have relied solely on manual analysis by researchers. The process of manually analysing a website to locate Dark Patterns is time-consuming and limits the number of websites that can be analysed. Using an automated approach we aim to collect data about Dark Patterns at a scale not possible with manual analysis. In addition, manual analysis relies on all researchers agreeing on and sticking to agreed criteria about what constitutes each Dark Pattern. Inevitably, human error results in mistakes and disagreement about the presence and prevalence of a Dark Pattern.

The main goal of this project is to design and implement a system that can automatically detect Dark Patterns on cookie dialogs. To achieve this goal the system first must be able to accurately collect cookie dialogs from the internet and save them for further analysis. Once a cookie dialog has been collected, the system must determine what clickable elements are present and the cookie setting behaviour of these elements. Using the information collected from a cookie dialog, the system then uses a rule-based approach to automatically infer the presence or absence of Dark Patterns.

The resulting system is capable of detecting 10 Dark Patterns automatically and optionally an additional 5 Dark Patterns can be manually inputted by the user via a user interface.

Despite its constraints, manual analysis is still essential to validate the accuracy of the system. Manual validation of the system was conducted on two smaller data sets of 500 websites each. Manual validation shows that the system was able to collect cookie dialogs with 98.8% accuracy. The system was then able to locate and classify clickable elements with an accuracy of between 95.1%-97.4% depending on the set of websites used. Finally, the system was able to detect Dark Patterns with an accuracy of between 99.0%-99.2%, again depending on the set of websites used.

After manual validation was completed, the system then automatically collected cookie dialogs on over 10,000 websites. This collection yielded a data set of over 2,000 cookie dialogs which we will use to answer the following questions:

- How common is it for websites to display a cookie dialog? What methods are the most effective at locating cookie dialogs on a web page?
- What clickable elements are present on cookie dialogs? How much effort does it take users to opt-out of cookies?
- What is the cookie setting behaviour of clickable elements on cookie dialogs such as the opt-in, opt-out and close buttons?
- What are the most prevalent Dark Patterns found on cookie dialogs?

From a sample of 10,000 websites, this study found that 20.92% of websites displayed a cookie dialog and that 89.5% of websites with a cookie dialog have at least one Dark Pattern present.

In summary, the main contributions of this project are as follows:

- A new ranking-based approach to locate cookie dialogs, which shows improved accuracy compared to previous studies.
- Design and implementation of a system that can automatically detect Dark Patterns on a cookie dialog.
- A taxonomy of Dark Patterns commonly found on cookie dialogs.
- 3 data sets of cookie dialogs and Dark Patterns collected using the system:
    1. A smaller set of the Top 500 most-visited websites of 2021 which have been manually validated.
    2. Another, smaller set of 500 randomly selected websites which have been manually validated. This was collected to ensure the system is not biased towards the most popular websites.
    3. A larger set of 10,000 randomly selected websites.

Chapter 2

Background

2.1 Cookies

HTTP Cookies (or simply just **Cookies**) are small pieces of data created by web browsers to allow websites to keep track of their user's activity. Cookies were first added in 1994 [50] into the Netscape Navigator browser and have since been implemented by all major web browsers. The name is derived from the computing term 'Magic Cookie' [29] which refers to a packet of data that is sent out and received back by your computer without being altered.

Cookies are used as part of the Hypertext Transfer Protocol (HTTP) [38], an application layer protocol that is the foundation of communication between users on the World Wide Web. HTTP is by itself a stateless protocol, which means that the web browser cannot retain any state from previous requests. Each time the user interacts with a web page (and triggers an HTTP request to be sent) the connection to the server will be terminated and the browser will create a new page without any knowledge of the user's previous activity. Cookies allow the web page to retain state by storing data locally in the web browser. Cookies are stored as key-value pairs with a number of additional attributes (see Cookie Attributes section for details).

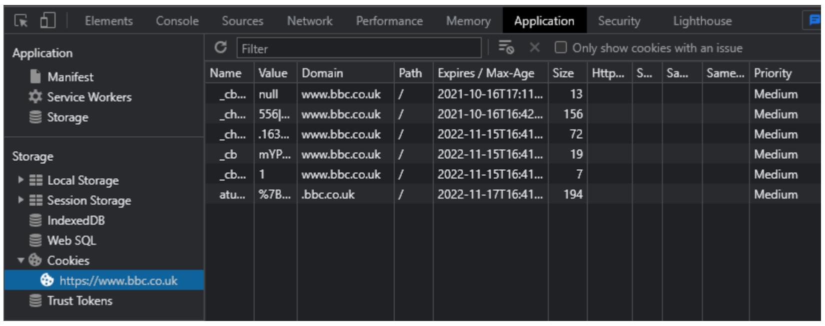
Cookies have a wide range of applications, some are required by the websites to keep track of state while others are used by third parties to track the activities of users on a web page. For example, many online shopping websites use cookies to keep track of what items you have in your shopping basket and without cookies, your basket would reset each time you interacted with the web page. We will define more rigid categories for classifying cookies in the Types of Cookies Section.

2.1.1 Cookie Attributes

An example of the Cookies that can be stored in the Chrome browser is shown in Figure 2.1. The implementation of cookies can vary slightly between web browsers but in most implementations cookies contain the following attributes [39]:

- **Name:** the name of the cookie.

- **Value:** the value/content of the cookie.
- **Domain:** specifies which hosts are allowed to receive the cookie.
- **Path:** specifies a URL that must exist in the request in order to send the header.
- **Expires/Max-Age:** specifies a time for when when Persistent cookies are deleted.
- **Size:** stores the number of characters the Name and Value attributes.
- **HTTPOnly:** if set to true makes the cookie inaccessible to JavaScript.
- **Secure:** specifies that encryption should be used to send the cookie to the server.
- **SameSite:** lets servers specify if cookies can be sent in cross-origin requests.



Name	Value	Domain	Path	Expires / Max-Age	Size	Http...	S...	Sa...	Same...	Priority
_cb...	null	www.bbc.co.uk	/	2021-10-16T17:11...	13					Medium
_ch...	556|...	www.bbc.co.uk	/	2021-10-16T16:42...	156					Medium
_ch...	.163...	www.bbc.co.uk	/	2022-11-15T16:41...	72					Medium
_cb	mYP...	www.bbc.co.uk	/	2022-11-15T16:41...	19					Medium
_cb...	1	www.bbc.co.uk	/	2022-11-15T16:41...	7					Medium
atu...	%7B...	.bbc.co.uk	/	2022-11-17T16:41...	194					Medium

Figure 2.1: Example of cookies stored on the Chrome Browser.

2.1.2 Types of Cookies

The common types of cookies are closely linked to the laws that cover them (see Legal Implications). A good set of categories for defining cookies are laid out by the EU guidance on how cookies are covered under GDPR [32]. This sets out three main ways of classifying cookies: by Duration, by Provenance and by Purpose. Not all cookies will fit into these three main categories and some cookies may fit into multiple categories at the same time. These three categories should cover the vast majority of cookies.

By Duration: Using the Expires/Max-Age attribute of a cookie we can determine its maximum lifetime, with this information cookies can be classified as one of the following:

- **Session cookies** if the Expires/Max-Age attribute is not set these cookies are used only while navigating a website. They are stored in memory and are never written to the hard drive and when the session ends they are automatically deleted.
- **Persistent/tracking cookies** if the Expires/Max-Age is set, these cookies remain on a computer until their expiration date is exceeded and are they are automatically removed when that date is reached.

By Provenance: Using the domain attribute of a cookie we can determine its origin, in terms of which website the cookie came from. We can then classify cookies as one of the following:

- **First-party cookies** originate from the website the user is currently browsing.
- **Third-party cookies** are generated by websites that are different from the web page the user is currently browsing. Analytics companies use Third-party cookies to track a user's activity across the web on any sites that contain their ads.

By Purpose: Cookies can also be classified by the purpose they are intended to serve. It is harder to do this as cookies are used for a wide range of purposes and it is up to each website on how to implement cookies. To allow legal constraints to be placed on how cookies are stored, GDPR guidance [29] sets out 4 different categories allowing us to effectively group cookies with similar purposes:

- **Strictly necessary cookies** are cookies that are essential in allowing users to browse the website and use its basic features without issue.
- **Preference/Functionality cookies** are used by websites to store choices made while browsing the website in the past.
- **Analytics/Statistics cookies** are cookies used by the website to record information about how users interacted with the website. These cookies are not meant to be able to identify a user and their purpose is to provide information to improve the website.
- **Marketing cookies** are cookies used to track your browsing activities and are used by advertising companies to help target users with more relevant ads. They are often classified as being third-party in origin and being persistent in duration.

2.1.3 Cookie Dialogs

A **cookie dialog** (or **cookie privacy warning**) is a notice found on a website to inform the user of the website's cookie policy. The notice will state that cookies or tracking tools are being used. The dialog may allow the user to accept, reject or select a level of cookie preference. It may also prompt a user to view a more detailed version of the website's cookie policy. Cookie dialogs first began to appear in 2002 when the EU's ePrivacy directive came into force [12]. They have become increasingly common following the introduction of the GDPR in 2018.

For the purposes of this project, we will define a **Clickable** as a web page element that can be interacted with and can be clicked on by the user's cursor. This element must be present inside a cookie dialog. A full list of clickable types is provided in the Clickable Location and Classification section.

2.2 Legal Implications

We will be considering the **Privacy and Electronic Communications Regulations (PECR)** and the **Data Protection Act 2018 (DPA)** [43] as these UK legislations are most relevant to cookies. If a website stores or accesses information stored on a user's device then it must comply with PECR first. The DPA then applies to any processing of users' personal information by websites outside of this storage.

2.2.1 PECR

PECR was brought into law to give people specific privacy rights in relation to electronic communications [44]. It is derived from the European Directive 2002/58/EC (or "the e-privacy Directive"). The rules in PECR relating to cookies are that a website must:

- Inform users the cookies are present.
- Explain the purpose and necessity of each cookie.
- Obtain consent before storing a cookie on a user's browser. This must involve some form of ”unambiguous positive action” [42] such as ticking a box.

There are exceptions for cookies that are essential for the website to provide its services. Such as remembering what items are in a user's online shopping basket, or ensuring security when logging in to an online bank.

2.2.2 GDPR

DPA is the UK's implementation of the EU's GDPR. It was brought into law in May 2018 and made sweeping changes to the previous version of the DPA 1998. It greatly expanded requirements relating to the handling of personal information by websites. It also focused on ensuring that users have access rights to their data [43].

The EU recommends that in order to comply with GDPR, websites must display a cookie dialog to allow users to control their cookie preferences [32]. This dialog must:

- Ask users for consent before placing any cookies.
- Provide enough information and sufficient options to allow users to make an informed choice.
- Allow users to decline the storing of non-essential cookies and then must allow the user access to all features of the website unhindered.
- Enable users to later change or withdraw their consent.

In many instances, it could be argued that cookie dialogs do not meet the minimum standards of GDPR. Some websites do not provide enough options for users to select a suitable preference. Other websites use interface design choices on cookie dialogs to push users towards accepting all cookies.

2.3 Dark Patterns

Dark Patterns are features of a website's interface designed to deceive users into accepting a choice that is not in their best interests [3]. The term was first proposed by user interface expert Harry Brignull in 2010 [3]. Dark Patterns are more than just bad design choices, they are deliberately crafted and maliciously designed to deceive users. In the context of cookies, Dark Patterns are design choices present on cookie dialogs intended to push users towards sharing more of their data [16]. They do not necessarily have to be visual design features and can also include hidden features such as the cookie setting behaviour of cookie dialogs.

Many of the examples provided by Brignull are specific to a particular user interface type and many do not apply to cookie dialogs at all. Using the examples of Dark Patterns provided by Brignull [3] as a baseline, Gray et al. [27] went on to refine these examples into categories, which are summarised in Table 2.1. These new categories subsume the examples by Brignull and each category attempts to group Dark Patterns with similar design motivations. For example, a common Dark Pattern on cookie dialogs is ”Only

Opt-in option is present on initial Cookie Dialog". This falls under the "Forced Action" category as the user is left with no choice but to accept all cookies.

To ensure that we cover a wide range of designer strategies, we will look for instances of at least one Dark Pattern in each of these categories. In the Dark Pattern Detection section we will go into detail on the Dark Patterns that the system supports. The reader may find it helpful to see some examples of Dark Patterns on cookie dialogs these are included in the appendix.

Category	**Description**
Nagging	A minor redirection of expected functionality that may persist over one or more interactions. Nagging often manifests as a repeated intrusion during normal interaction, where the user's desired task is interrupted one or more times by other tasks not directly related to the one the user is focusing on.
Obstruction	Impeding a task flow, making interaction more difficult than it inherently needs to be with the intent to dissuade an action. Obstruction often manifests as a major barrier to a particular task that the user may want to accomplish.
Sneaking	An attempt to hide, disguise, or delay the divulging of information that has relevance to the user. Sneaking often occurs in order to make the user perform an action they may object to if they had knowledge of it.
Interface Interference	Any manipulation of the user interface that privileges specific actions over others, thereby confusing the user or limiting discoverability of important action possibilities. Interface interference manifests as numerous individual visual and interactive deceptions.
Forced Action	Any situation in which users are required to perform a specific action to access (or continue to access) specific functionality. This action may manifest as a required step to complete a process or may appear disguised as an option that the user will greatly benefit from.

Table 2.1: Dark Pattern Categories as defined by Gray et al. [27].

2.4 Related Work

There have been numerous studies [13][18] into the impact that the introduction of the GDPR has had on the prevalence of cookie dialogs.

Degeling et al. [13] conducted a study in 2018 into the changes to the Top 500 websites from each of the 28 EU member states (as of 2021, 27 member states and the UK). They recorded the changes each month from January to October 2018. From January 2018 to May 2018 (when the GDPR was introduced) they observed a 16% increase in the number of websites displaying cookie dialogs. They also observed a slight decline in the number of websites using tracking cookie libraries and a small increase in the number of websites asking for explicit consent before using these libraries. They also analysed the cookie dialogs for the requirements set out by the GDPR. They found that many websites do not offer their users much choice when it comes to cookies. They found that the right to withdraw consent was limited by the same-origin policy which restricts websites from being able to withdraw consent from third-party cookies. In terms of which cookies require consent and those covered by "legitimate interest", they found regulators needed to provide clearer guidelines. They concluded that overall the GDPR has had a positive impact on web privacy and transparency but also acknowledged that many websites still do not meet the minimum requirements set out by the GDPR.

Matte et al. [18] conducted their study using the Top 1000 websites for each EU country and Top Level Domain from the Tranco rankings [19]. They noted the main benefit of using Tranco rankings was that it used an aggregation of several rankings to overcome flaws in the other lists such as the website being unreachable. They also excluded the Cisco Umbrella list from the Tranco ranking as they noted this is based on DNS which may not be representative of traffic to a website. It is definitely worth considering these factors when deciding what website list to use for this project. In their study, Matte et al. [18] looked to see if websites respected the option chosen by the user on the cookie dialog and found that 54% of websites do not respect this choice to some extent. They also studied each cookie dialog they collected finding that 10% of websites store consent before a choice is made, 7% of websites provide no way to opt-out and 47% of websites have a choice pre-selected.

Englehardt et al. [15] created an automated tool using the Open Web Privacy Measurement Framework (OpenWPM). Using this framework they were able to analyse multiple websites simultaneously which allowed them to conduct a study on over 1 million websites. They found that just 123 of the 81,000 third-party cookie providers are present on more than 1% of the websites. Meaning that a small number of companies make up the majority of third-party tracking cookie providers with Google being the most prevalent owning 12 of the top 20 third-party providers. Using categories of websites from the Alexa ranking [1], they found that news websites have the highest average number of tracking cookies present. Many studies [47][14][18][15] chose to use the Selenium browser automation tool due to it being supported by multiple browsers. Englehardt et al. [15] used Selenium as they found that many websites did not load sites correctly when using a headless browser. A headless browser is a web browser that does not have any graphical user interface, it is typically quite easy for websites to tell when they are being visited from a headless browser. They also found that using a PhantomJS (a headless browser) meant that many website technologies and browser extensions were not supported.

Over the past few years, there have been a few studies [14][47] that have created tools to automatically detect cookie dialogs on a website.

Eijik et al. [14] conducted a study using OpenWPM to automatically detect cookie dialogs on 1800 websites using the ”I don’t care about cookies” CSS selector list. Using manual validation to check their results, they found cookie dialogs present on 40.2% of websites and were able to detect this correctly 91% of the time. However, the accuracy of this approach however relies solely upon the CSS selector list itself. Many websites regularly change the CSS selector of cookie dialogs to prevent CSS selector lists from being accurate. So if the list is poorly maintained then it will likely not pick up many cookie dialogs.

Petronyte [47] worked to create a tool to automatically detect cookie dialogs with improved accuracy. They used the CSS Selector approach discussed above to detect cookie dialogs and then checked that at least one n-gram was present on the cookie dialog. They generated the n-grams from around 500 sample cookie dialogs. If the CSS Selector method did not find any dialogs they added a keyword search to attempt to find a cookie dialog in the page HTML. On the Top 1000 websites from the Tranco [19] rankings, they found cookie dialogs present on 54.3% of websites and automatically detected 98% of cookie dialogs.

Most research in the area of Dark Patterns has focused on other types of user interfaces other than cookie dialogs. For the purposes of this project, it will be necessary to create a list of Dark Pattern types based on those which appear commonly on cookie dialogs. This will allow us to classify a cookie dialog based on the type(s) of Dark Pattern it contains. In turn, this will allow us to get a good sense of the overall prevalence of the different types of Dark Patterns on cookie dialogs across the internet. A good starting point to create this list was to look at the original types of Dark patterns defined by Brignull on his website [5]. They also have a Twitter page [4] that contains a huge collection of examples of Dark Patterns from a wide range of user interfaces. There are several papers [27][17] that use the original types of Dark Pattern from Brignull as an inspiration for how to categorise Dark Patterns.

Gray et al. [27] manually collected a corpus of Dark Patterns from a broad range of website interfaces and then provided an overview of the different categories that these Dark Patterns typically fall into. They developed these categories based on the original types of Dark Patterns provided by Brignull [5] and focused on explaining these categories in depth. However, they had no focus on how prevalent Dark Patterns were so it is impossible to get a sense of how common each category is on the web. Their approach to locating these Dark Patterns was to employ two researchers to search the web for Dark Patterns which took them a long period of time (two months) and they also appear to have used no form of tools (except a web browser) to help them locate these Dark Patterns. This meant their work relied solely on the skill of these researchers and they provided little explanation on how exactly they found these examples beyond simply browsing the web.

Machuletz et al. [17] conducted a user study with mock cookie dialogs used to see if the presence of a Dark Pattern would cause users to consent to Cookies more often. The first Dark Pattern they looked at was the ”highlighted default button that selects all purposes”. They found that with this Dark Pattern present users are more likely to consent, regret this decision more and perceive the website as more deceptive. They also looked at the Dark Pattern ”multiple purposes given for the website to store cookies” and found that users it took more time and effort for users to make a choice.

Krisam et al. [16] conducted research into the prevalence of Dark Patterns on cookie dialogs on the top 500 German websites, ranked using the Alexa [1] rankings. They developed a set of categories based on those original examples from Brignull [5] but narrowed down the categories to be specific to cookie dialogs. They relied mainly on a manual process to tag Dark Patterns present on cookie dialogs. The only automation they added was to create a tool to screenshot cookie dialogs but they did not discuss how this was achieved.

Chapter 3

Design & Implementation

Our overall goal is to build a system that can automatically detect Dark Patterns on cookie dialogs. To detect Dark Patterns we must first collect information about the cookie dialog itself. The two main types of information we need about a cookie dialog are the clickable elements present and the cookie setting behaviour of these elements. This introduces a chain of dependencies. First, to determine the cookie setting behaviour of clickables, we need to locate clickables. Second, to locate these clickables, we must first find a cookie dialog itself. Finally, to find a cookie dialog, we must be able to automatically interact with a web page and locate elements.

3.1 Requirements Elicitation

Keeping the chain of dependencies in mind we can elicit each of the functional requirements our system must have and the order in which they must be developed. This gives us the following functional requirements which specify that the system must:

1. Be able to automatically interact with a web browser and locate elements present on a web page.
2. Be able to find cookie dialogs on a web page.
3. Be able to locate and classify any clickables.
4. Be able determine the cookie setting behaviour of the dialog.
5. Be able to infer the Dark Patterns present on a cookie dialog.
6. Have a UI to allow manual validation and accuracy to be calculated.

Breaking the system down into this chain of dependencies allows us to implement each of the requirements in order. This allows us to thoroughly test each component individually and with the components it requires. Breaking this system down into manageable modules will help to reduce the complexity and improve the reliability of the system. To judge the operation of the system it must also meet the following non-functional requirements:

1. Accurate: the system must have a good level of accuracy for finding a dialog, locating clickables and detecting Dark Patterns. We hope for an accuracy of 90% for each aspect but strive to make the system as accurate as possible.

2. Internationalisation: the system should be able to operate in a range of written languages, not just English.
3. Portability: the system should be easy to install and compatible with different operating systems. The data collected should also be stored in a format that can be exported for further research.
4. Robust: the system must be able to cope with failures during execution, both those caused by software faults and network connection issues. The software should store all data on disk as soon as possible and be able to restart the process if required.
5. Timely: the system must be able to analyse a website within a reasonable timescale or it could be argued that there is no benefit over manual inspection. We propose that 30 minutes is a good upper limit for the system to do a full analysis of a website as anything above this a human would be consistently quicker.

3.2 Website Scraping

We use the Selenium web scraping library in python. Specifically, we used the chromedriver (version 97.0.4692.71) in Selenium to allow us to interact with websites in Chrome. We used the Chrome browser as it should get us the most realistic representation of what the typical user's browsing experience is like. This is because Chrome is the most widely used browser [51] and has fewer default privacy settings compared to other Browsers such as Firefox [40]. As websites can change based on which country the user is in we also use a VPN to ensure a constant location. During all operations of our system, a UK based VPN server was used.

3.2.1 Handling Website Connections

To avoid having multiple subdomains from a website, the system takes the domain of a website in the format: `domain.suffix` (where the domain is the Second-Level domain name and suffix is the Top-Level domain name). Conveniently this is the same format used by the Tranco list. However, to connect to a Website using ChromeDriver we must give it a complete URL. Not all websites use the HTTPS protocol or have URLs starting with '`www.`', so we need to make multiple attempts to connect to a website in the following order:

1. https://www.domain.suffix
2. http://www.domain.suffix
3. https://domain.suffix
4. http://domain.suffix

If we fail to connect to a website after all 4 attempts then the system considers the website unreachable and the failure to connect is recorded in the database.

The Tranco list attempts to filter out websites that are not meant for humans. However, it is not always successful. In these cases, if the website cannot be reached or entering the domain takes us to a page that does not exist then it will be treated as an unsuccessful connection. In some cases, a blank or error message can be displayed in the form of

HTML. In this case, the page is considered a valid page and we will check for a cookie dialog (it is very unlikely one will be found).

Some websites use CAPTCHA [52] tests to distinguish real users from robots. It would be technically challenging and unethical to automate the process of bypassing this test. No CAPTCHA tests were encountered during manual validation of the system so we do not foresee this being a major issue during automated testing.

3.2.2 Cookie Collection

To assess the cookie setting behaviour of the dialog we must be able to collect all the cookies set by the web page at a particular time.

As was discovered by Molar [36] in their study, Selenium provides an API to collect cookies but it is not effective in retrieving all cookies. It misses out on many important cookies including the majority of third-party cookies.

The best way to accurately retrieve all cookies is the Chrome DevTools Protocol command `Network.getAllCookies` [24]. This returns all cookies currently stored by the Chrome browser. It was observed during development that it took up to 30 seconds for all cookies to be set. So to ensure all cookies have been set, the system waits 30 seconds. After 30 seconds, the system compares the cookies every 10 seconds until there are no changes in the cookies set and only then do we collect cookies.

To ensure that there is no cross-contamination between cookie collections we must ensure that the Chrome Browser cache is cleared after each cookie collection. The system automatically navigates to the `chrome://settings/clearBrowserData` page on the Chrome browser and clicks the 'Clear Data' button. This is the same method a user would take to clear their browsing history [25] so should be as effective as possible.

3.2.3 iframes

An Inline Frame or `iframe` is a special type of HTML element that allows you to embed various types of interactive media within a web page [37], essentially allowing us to have a web page within a web page. Selenium handles iframes different to other elements and a context switch is required to interact with an iframe. Many websites use iframes to store their cookie dialogs so we must ensure our system can handle the interaction with them. It is important to note that there can be many instances of iframes on a web page and that they can be used to store a wide range of media, not just cookie dialogs, so we cannot simply infer that every iframe is a valid cookie dialog.

To locate all iframes on a web page we can simply search for all elements on the web page with the HTML tag name `iframe`. This will return us a list of all iframes. Now we can interact with each iframe individually. To do this we need to context switch to the iframe through the `switch_to.frame` API that Selenium provides [10]. Now we can perform any required actions, such as retrieving HTML or taking a screenshot of the iframe. Finally, to return to the main body of the web page, we need to context switch again. This time we use the `switch_to.default_content` API that Selenium provides [10]. We will discuss the specific uses of iframes in our system further in the following sections.

3.2.4 Google Translate API

To allow our system to be compatible with a range of different languages we use the Python Google Translate API [28]. This provides the same translations as the Google Translate website [26]. Google Translate was chosen as it supports a wide range of languages and their Python API is well tested. Using this API allows us to detect the language and translate any text into English. It should be acknowledged that using any Online Translate API will not always give us the grammatically correct translation. However, for this project, we are generally only interested in checking for the presence of keywords in any translated text. So this accuracy of translation will suffice. We will discuss the specific uses of the Google Translate API in our system further in the following sections.

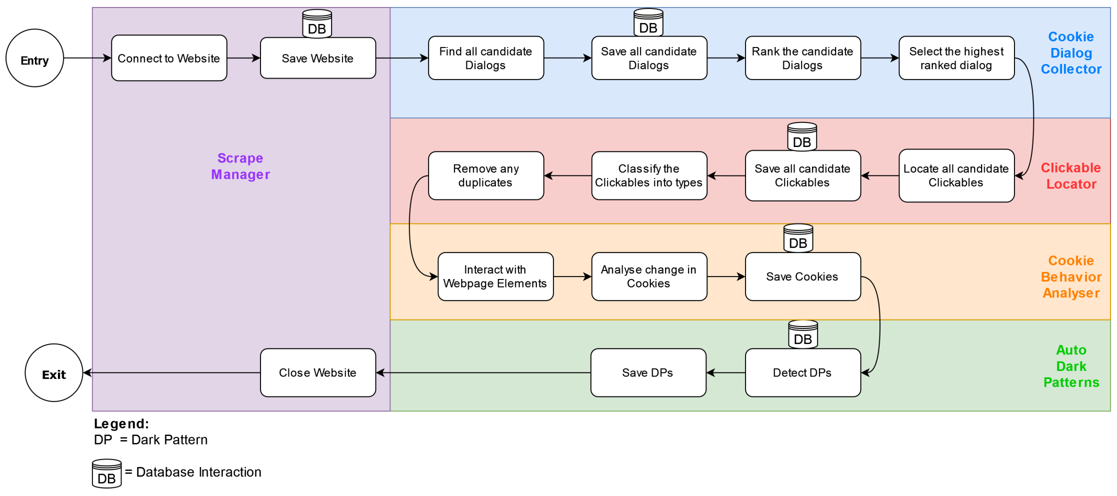
3.3 System Design

To achieve our goal of automatically detecting Dark Patterns, the system has been decomposed into 4 modules that each provide a crucial piece of functionality. Figure 3.1 shows the main steps in each module and the workflow of the system. The rest of this chapter will provide a breakdown of these modules which are as follows:

1. **Cookie Dialog Collector:** finds the cookie dialog (if one is present).
2. **Clickable Locator:** if a dialog was found then, this module locates any clickables on the Dialog and classifies them into types.
3. **Cookie Behaviour Analyser:** this module analyses the Cookie Setting Behaviour of the dialog to allow us to infer some further Dark Patterns.
4. **Auto Dark Patterns:** the module uses the data collected from the Dialog and Clickables to detect any Dark Patterns present on the Dialog.

The **Scrape Manager** module is responsible for the initialisation of web pages and the control of interactions between modules in the system, shown in Figure 3.1. This module abstracts the process of analysing a cookie dialog for Dark Patterns and simply allows the user to pass in a list of website domains which they want analysed. It allows configuration options to be specified, the main option that needs to be specified is the Mode of Operation.

All the data collected from the system will be stored in a SQLite database to ensure it is not lost if there is a system failure. SQLite is a server-less database engine this reduces the complexity and installation time of our system. The `.db` file format SQLite provides can be easily exported which conforms with our Portability non-functional requirement. Interactions with the database are shown in Figure 3.1 and these also act as restart points should the system terminate unexpectedly. The database schema and description of the exact data collected are included in the appendix.



Entry
Connect to Website
Save Website
DB
Find all candidate Dialogs
Save all candidate Dialogs
Rank the candidate Dialogs
Select the highest ranked dialog
Cookie Dialog Collector
Scrape Manager
Remove any duplicates
Classify the Clickables into types
Save all candidate Clickables
Locate all candidate Clickables
Clickable Locator
Interact with Webpage Elements
Analyse change in Cookies
Save Cookies
Cookie Behavior Analyser
Exit
Close Website
Save DPs
Detect DPs
Auto Dark Patterns
Legend:
DP = Dark Pattern
DB = Database Interaction

Figure 3.1: Diagram showing the main actions taken to analyse a single website.

3.3.1 Modes of Operation

Using manual validation to correct mistakes made by an automated system will lead to a higher level of accuracy. However, manual inspection of websites is a time-consuming process and conducting manual validation for a large set of websites would be infeasible. Therefore, we require two modes of operation:

Auto Mode: is the fully autonomous version of the system that operates without user interaction to analyse a website. This was developed to allow me to conduct a large scale scrape of websites that would be infeasible with user interaction involved. The process taken by this mode is shown in the Figure 3.1.

Manual Mode: is the version of the system that allows the user to validate the choices that the system makes. This was the initial version of the system that was used to assess the accuracy of the system during manual validation. This mode follows all the same steps as shown in the Figure 3.1. Additionally, via a user interface, manual mode allows the user to:

1. Validate that the highest-ranked dialog is correct.
2. Validate that all clickables have been found and classified correctly.
3. Validate that any automatically detected Dark Patterns are correct.
4. Prompts the user to reject cookies by interacting with the dialog. This allows a comparison between the cookies before and after rejection. The number of clicks the user makes is recorded.
5. Input any additional Manual Dark Patterns that they observed.

Although the central objective of this project is to develop an automated tool to detect Dark Patterns (which is what Auto Mode provides), it was important to have the Manual Mode as it allows the accuracy of the system to be tested. It also allows a further set of Manual Dark Patterns to be inputted by a user. These are Dark Patterns that were identified as being present on cookie dialogs but were deemed too complex to be detected automatically within the scope of this project. We will discuss these further in the Dark Pattern Detection section.

3.3.2 Dialog Collection

Cookie dialogs come in all shapes and sizes, so there is no trivial approach to locating all cookie dialogs. Many websites purposely make it difficult to automatically detect cookie dialogs.This is likely to ensure that ad-blockers cannot effectively block cookie dialogs.

It is important to note that websites can have more than one cookie dialog (an example of this is shown for Dark Pattern 7). As we will explain further in the Dark Pattern Detection section, the appearance of more than one cookie dialog is considered a Dark Pattern. So for this project, we will assume that websites can have more than one cookie dialog. However, for the **Clickable Locator** and **Cookie Behaviour Analyser** modules to work correctly, we must select one of these dialogs to be valid. So we will consider our system successful if it manages to locate both dialogs as candidate dialogs and the ranking-process ranks either one of these dialogs first. This will allow us in the **Auto Dark Patterns** module to consider if there is more than one cookie dialog.

CSS selectors are patterns used to specify which HTML elements on a web page we want to select. In a previous study B. Molnar [36] identified a method to use Ad-Blocker Cookie CSS Selectors Lists to identify cookie dialogs. These lists are publicly available and updated regularly and contain CSS selectors which identify cookie dialog elements. The website filterlists.com [7] provides an overview of the available CSS selector lists and we can apply a filter to only view cookie-specific CSS selectors lists. From this website we can see that the two most widely used and frequently updated lists are **EasyList** [46] and **I don't care about cookies** [31] so these lists were used to update CSS selectors in the system. Both these lists are used by major ad-blockers including Adblock Plus, AdBlock and uBlock Origin [46].

The most common tactic that websites use to make cookie dialog detection difficult is the obfuscation of CSS selectors on cookie dialogs. This is when they use choose CSS selectors on dialogs that have nothing to do with cookie dialogs. This makes it harder to guess and reduces the chance that it will appear in a CSS Selector list. Even if this CSS selector is then added to a Cookie CSS selector list, websites will frequently regenerate a new pseudo-random CSS selector which will make the old one obsolete. The updating of CSS selectors effectively creates a never-ending race for ad-blockers lists to stay in sync with websites. This leads to the efficacy of ad-blocker lists being based on how frequently they are updated.

An extreme case of updating CSS Selectors that we discovered was google.com.hk, the landing page for Google Search in Hong Kong. This web page randomly regenerates the CSS selector on their cookie dialog for each new user that visits the page. Meaning there is no effective way for ad-blocker lists to be kept in sync with this web page.

For these reasons, we cannot solely rely on ad-blocker lists for accurate cookie dialog detection and we must look at other features of web pages to correctly identify cookie dialogs. We will discuss this more in the Ranking all the Candidate Dialogs section.

The process of finding a cookie dialog on a web page can be broken down further into the following steps which we will discuss in details in the following sections:

1. Find all Candidate Dialogs, all the potential elements on the web page that could be cookie dialogs.
2. Rank all the Candidate Dialogs.
3. Select the highest-ranked Candidate Dialog as the correct cookie dialog.

3.3.2.1 Finding all Candidate Dialogs

In this stage we attempt to identify as many possible candidate cookie dialogs, this allows the candidate ranking process to have a good range of elements to work with. The objective of this stage is to have as high recall as possible and then allow the ranking process to make a precise judgement if any candidates are suitable representations of a cookie dialog.

The features captured for each candidate dialog are a screenshot (this also gives us the size and position of the dialog), the text, the HTML content and the method used to locate the dialog. All these features are vital in allowing the ranking process to assess the suitability of each candidate. Taking a screenshot of each candidate is particularly important as Selenium only allows you to screenshot elements that are visible to the user. This allows us to filter out many invalid candidates during the ranking process.

To locate elements on a web page that could be cookie dialogs we use the following methods:

1. **Domain-Specific CSS Selectors:** these are CSS selectors which are only valid for identifying a cookie dialog on a particular domain.
2. **iframes:** are an HTML element tags that are used to embed an external element in the current web page. Cookie dialogs are commonly contained within iframes.
3. **General CSS Selectors:** these are CSS selectors which are commonly found in cookie dialogs.
4. **Custom CSS selectors:** these are additional CSS selectors developed as part of the system implementation to search div tags containing common terms. The full list of Custom CSS Selectors is included in the appendix.

Previous studies by Petronyte [47] and Molnar [36] considered elements found using General CSS selectors only if they found no elements using Domain-Specific CSS Selectors. By collecting candidates using Domain-Specific, General, iframes and Custom CSS selectors first we aim to locate all possible candidates dialogs. This will ensure our system has a very high recall, meaning our ranking process will have a large number of candidates to assess. Then during the ranking process, we will use the range of features collected about each candidate to precisely predict which candidate is the best possible representation of a cookie dialog.

Our system contains around 19,000 general and 13,000 website-specific CSS selectors in total. During implementation, we discovered that Selenium was very slow for searching through such a large list of CSS selectors. Depending on the complexity of the web page, we found it took up to 20 minutes to search the whole list which risks not being acceptable per our Timely non-functional requirement. So instead we use the Python `Beautiful Soup` [48] library to search the HTML content of the web page as it is much more efficient and gave us a maximum search time of approximately 5 minutes.

As we discussed in the iframes section, iframes are handled differently than other candidate dialogs. So we first we must perform a context switch. Then we can collect a screenshot, HTML content and text from the iframe body. To ensure that the `Clickable Locator` and `Cookie Behaviour Analyser` modules can interact with any clickables on this candidate, we also record the method used to locate the candidate was an iframe. We will need to perform the same context switching to interact with any clickables on an iframe.

3.3.2.2 Ranking all the Candidate Dialogs

During the early stages of development, we observed that CSS selectors alone were not always able to accurately identify cookie dialogs and would often return false positives. So to improve the accuracy of the system we designed a new ranking-based approach. This ranking process considers a range of features collected about each candidate. Features captured for each candidate dialog include collection method, text content, outer HTML and a screenshot. Using a combination of factors derived from these features we can assign a score to each candidate. All candidates start with a score of 0 and the higher the score the more suitable a candidate is. Candidates can gain scores, lose scores or are removed from the ranking process completely based on a range of factors summarised in Table 3.1. Candidates with a score of 0 or less are deemed to not represent a cookie dialog and are disregarded after the ranking process is complete.

Candidate Feature	**Ranking Factor**	**Weight**
Collection method	Found using Domain-Specific CSS selector	+10
	Found using General CSS selector	+5

Text Content	Contains common cookie dialog N-grams	unigram = +1
		bigram = +2
		...

	Length less than 5	-20
	Length greater than average plus 100	-20
	Contains no text	-100
Outer HTML	Substring of another candidate	-1
	Same as another candidate	Removed
Screenshot	Candidate could not been screenshotted	Removed

Table 3.1: Table showing the factors and weights used to rank candidates.

Weights have been assigned to each factor based on their importance of indicating whether a candidate is a valid cookie dialog or not. In the following paragraphs we will go into more details about each factor and justify the weights assigned to them.

Firstly, we can look at the collection method of the candidate which allows us to differentiate between the ways a candidate can be located. If the candidate was **Found using a Domain-Specific CSS Selector** or **Found using a General CSS selector** this is more likely (but not guaranteed) to identify a cookie dialog due to it being from a publicly validated list. Since Domain-Specific CSS Selectors specify the exact website, they were deemed to be more important than those found using General CSS selectors and are assigned a higher weight. Elements found using our custom list of CSS selectors are not deemed to be any more important than other elements as they have not been publicly validated. iframes are also not assessed to be any more important as there is a wide range of content that can be contained within an iframe not just cookie dialogs.

Secondly, we can compare the text content of the candidate to text commonly found cookie dialogs. A good way of doing this is to check whether or not the text **contains N-grams commonly found in a cookie dialog**. From the text of 100 manually collected cookie dialogs, a set of N-grams was generated up to the 5-gram level. These were manually curated to ensure that they only contained N-grams which help identify a cookie dialog from other elements. The full list is included in the appendix. For each N-gram found in the candidate's text, a score is added based on the length of the N-gram. Longer N-grams are considered to be more important at identifying a cookie dialog. For example, the uni-gram 'cookie' is less important than the tri-gram 'we use cookies' at identifying the dialog as the uni-gram is a sub-string of the tri-gram. Since all our N-grams were generated in the English language we must first translate any non-English text. We can detect the language of and translate any text using the method discussed in the Google Translate API section. The performance of this factor could be increased by generating N-grams from a larger set of cookie dialogs.

Another important aspect of the text content is the total number of words it contains. Elements with text **Length less than 5** words are generally found to be clickable elements rather than cookie dialogs so are given a negative weighting to lower their rank. Candidates which **contain no text** were very unlikely to be a cookie dialog so were afforded a very large negative weighting. Looking at the average number of words across all candidates allowed me to assess whether a dialog was exceptional long. We initially tried lowering the score of candidates with more than the average number of words but in many cases, the correct dialog was just above the average number of words. So we decided to only lower the score if a candidate has a **Length greater than average plus 100**. In many cases, candidates which had an exceptionally large number of words did contain the cookie dialog but also included other non-relevant content.

Thirdly, another feature we can look at is the outer HTML content of the candidate. The Outer HTML is a string made up of the HTML element itself, including its attributes, start tag, and end tag. By checking this string we can assess the similarity of candidates to each other in a more precise way than comparing the text content. If a candidate's **Outer HTML is a sub-string of another candidate's Outer HTML** then the first candidate is a child of the second candidate. Child candidates are more likely to be clickable elements or incomplete parts of a cookie dialog. So to increase our chances of locating the whole cookie dialog we want the parent candidate to be ranked higher than any child candidates. To ensure that the parent element will be ranked higher than a child then a small negative weight was added.

We can also look at whether a **Candidate has the same HTML content as another dialog**. If this is the case, then we have likely selected the same element multiple times using different CSS selectors. We remove any duplicate candidates, giving preference to keeping candidates selected by Domain-Specific and General CSS selectors.

Finally, we look at whether a screenshot of the candidate using Selenium was possible. Selenium only allows you to screenshot elements that are visible to the user. If a candidate is not visible to the user then this is not likely a cookie dialog. So if a **Candidate could not be screenshotted** then we removed it from the list of possible candidates.

3.3.2.3 Selecting the highest-ranked Candidate Dialog

Once the ranking process is complete, we can then select the highest-ranked dialog with a positive score to become the valid dialog for this web page. The ranking process will give candidates that are not likely to be cookie dialogs negative scores. So if there were no positively ranked candidates then there should be no dialog present on the web page and we can stop our analysis of the current web page at this point. Finding a valid dialog will then allow us to begin the next phase of the process: Clickable Location & Classification.

3.3.3 Clickable Location & Classification

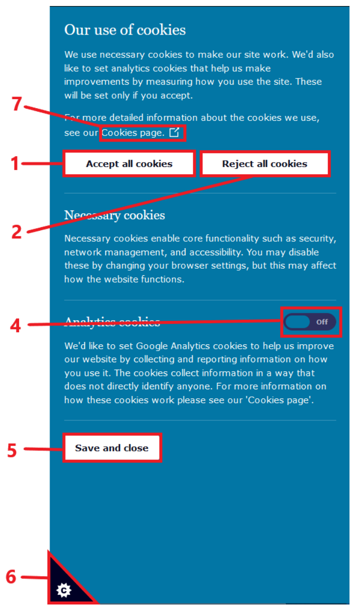
Once, we have located a valid cookie dialog we then want to find the clickable elements located on the cookie dialog. The process of locating clickables on a dialog can be broken down further into the following steps:

1. **Locate all Candidate Clickables**, that is all elements on the dialog which the user can interact with.
2. **Classify the clickables into types**, as specified in the list of clickables below.
3. **Remove any duplicate clickables**.

The list below contains types of clickables that a cookie dialog may contain:

1. **Opt-in option:** button which allows the user to consent to all Cookies. Usually worded affirmatively using words such as ’Accept’, ’Yes’ or ’OK’. However, depending on how the cookie dialog has phrased the wording of this button may be inverted with the Opt-out button.
2. **Opt-out option:** button which allows the user to reject the collection of Cookies depending on their preference. Usually worded negatively using words such as ’Reject’, ’No’ or ’Opt-Out’.
3. **More options:** button which redirects the user to another dialog where there are more preference options available.
4. **Preference Slider:** a single element that allows users to consent or reject certain types of cookies before confirming this choice using the Confirm Preferences button. We also distinguish between preference sliders that are enabled or disabled.
5. **Confirm Preferences:** a button used to confirm the cookies preference made using sliders. In some cases, the Opt-out button may take its place.
6. **Close option:** button that allows the user to close the cookie dialog without selecting an option.
7. **Policy Link:** link to the website’s Privacy or Cookie Policy page which does not contain any options to select cookie preference.
8. **Other:** any other non-relevant clickable on the dialog.
9. **Duplicate:** type used to specify the clickable is a duplicate of another clickable.

Figure 3.2 shows examples of Clickables on a cookie dialog.


Our use of cookies

We use necessary cookies to make our site work. We'd also like to set analytics cookies that help us make improvements by measuring how you use the site. These will be set only if you accept.

For more detailed information about the cookies we use, see our Cookies page.

Necessary cookies

Necessary cookies enable core functionality such as security, network management, and accessibility. You may disable these by changing your browser settings, but this may affect how the website functions.

Analytics cookies

We'd like to set Google Analytics cookies to help us improve our website by collecting and reporting information on how you use it. The cookies collect information in a way that does not directly identify anyone. For more information on how these cookies work please see our 'Cookies page'.

Figure 3.2: Examples of Clickables (highlighted in red) on a cookie dialog (Source: https://ico.org.uk/) Note: numbers correspond to the list of clickables.

In the following paragraphs, we will discuss each of the steps required to locate clickables in more depth.

Firstly, to **locate all the possible clickables** we use a custom set of CSS selectors (see the appendix for the full set). By appending the CSS selector of the cookie dialog to the start of the clickable CSS selectors, we ensure that we only select clickable elements which are located on the cookie dialog itself. Otherwise, we risk selecting every clickable on the web page which is not desirable. As we did with dialogs we can filter out any clickables for which a screenshot cannot be taken. This is since Selenium only allows elements visible to the user to be screenshotted. So if a clickable is not visible to the user then this is not likely a valid clickable.

Secondly, we **classify the type of each clickable** based on its text content. After the classification, each clickable will be assigned to exactly one type. To classify a clickable into a type we check for the presence of keywords in its text. If a clickable does not fall under any of these types then it is allocated to the 'other' type and deemed not useful. The keywords used for classification were derived from the study by Petronyte [47] who in turn derived their set of words from Molnar [36]. They have been expanded during the implementation of this system, the most notable difference is the inclusion of preference sliders in our set of clickable types which required their own set of keywords. The keywords for each type are shown in the appendix. Since all of our keywords were generated in English we must first translate any non-English text, we can do this by following the steps we discussed in the Google Translate API section. Clickables usually contain a small amount of text, so if we find a clickable that contains more than 10 words then it is likely we have selected another element. In this case, we consider the clickable invalid and assign it to the 'other' type.

To improve the accuracy of our classification we apply case folding and stemming to any text before classification. Stemming is a form of normalisation which attempts to reduce morphological variations of words to a common stem word. The system uses the `stemming.porter2` library [6]. This stemmer provides a good balance between efficiency and aggressiveness, appearing to not be too aggressive when stemming words and a good set of results are returned. Case folding is the replacement of all uppercase letters in a string with their lowercase counterparts. This serves to ensure that all instances of the same word are stored under a single entry in the index, regardless of the case of the letters. This removes the capitalisation of letters which increases the information retrieval performance and ensures that all relevant results are returned to the user.

During the implementation, it was observed that keywords alone were not enough to locate all of the 'close option' elements. So for this element we must take additional steps to locate all instances of this type. It is common for the close option to not contain any text content. So we need to use CSS selectors to find button types and we created custom set of CSS selectors based on close buttons observed during implementation, this set is included in the appendix.

We also observed that not all preference sliders could be located using keywords also due to many preference sliders not containing any text content. So we also added CSS selectors to locate Preference sliders, we only observed one additional CSS selector is used to identify preference sliders, which was `input[type="checkbox"]`.

Finally, now that we have located all candidate clickables we want to **remove any duplicate clickables**. Duplicates can occur when two CSS selectors locate the same element on a web page. To filter out duplicates we compare the OuterHTML content of candidate clickables. If any two clickables have the same HTML then we change the type of one of them to 'duplicate'. The 'duplicate' type is used to specify that the clickable should not be used for Dark Pattern analysis. This type of clickable does not serve us any further purpose during automatic analysis and is only kept for the purpose of manual validation.

3.3.4 Measuring the Behaviour of the Dialog

To allow us to infer the presence or absence of Dark Patterns 9 and 11 (we will describe these in the Dark Pattern Detection section) we need to conduct some additional cookie captures. For Dark Pattern 9, we need to assess the cookie setting behaviour of the 'Close button' clickable, only if this element is present on the dialog. For Dark Pattern 11 we need to assess 'Opt-in button' and 'Opt-out button' clickables, only if both of these elements are present on the dialog.

To avoid cross-contamination between cookies we need to clear the Browser cache. We can follow the same steps as described in the Cookie Collection section to achieve this. We then need to reload the web page so that the cookie dialog appears again in its original state. Once the website has loaded we can then interact with the desired element on the validated dialog using the CSS selector we collected for the clickable element. We do this by calling the Selenium `click` method [9] to simulate a user click on the element. We can then collect cookies present after this click, again following the steps described in the Cookie Collection section.

There is always a small chance that the web page does not load the cookie dialog correctly or that the web page may change CSS selectors dynamically between visits. To ensure we are interacting with the same cookie dialog as before, we compare the HTML content of the current dialog with the dialog stored in the database from the Dialog Collection stage. Any failure during this stage is noted in the database and the cookie interaction results are deemed invalid.

3.3.5 Dark Pattern Detection

Now that we have located a valid cookie dialog, found all clickables, and measured the cookie setting behaviour we can proceed with our main objective of detecting Dark patterns. The exact method of detection varies between each Dark Pattern. Examples of Dark Patterns located by the system automatically are provided in the appendix.

The main goal of this project is to automatically detect as many Dark Patterns as possible. However, it became clear that some of the Dark Patterns commonly found on cookie dialogs were very difficult to detect automatically and human judgement was sometimes required.

This left us with two detection types:

1. **Automatic** detection where our system will automatically infer the presence or absence of the Dark Pattern based on the criteria.
2. **Manual** detection where the user must manually input the Dark Pattern.

Note that the detection types roughly correspond to the Modes of Operation the system has. The Automated mode of operation only detects Automatic Dark Patterns, whereas the Manual mode of operation allows both Manual and Automatic Dark Patterns to be inputted by the user, via the user interface.

To ensure we could implement automatic detection for as many Dark Patterns as possible, we ranked the set of Dark Patterns we had from easiest to hardest detect. This allowed us to begin implementing the easiest Dark Patterns first and resulted in the system being able to detect a total of 10 Dark Patterns automatically.

In the remainder of this chapter, we will provide a taxonomy of all the Dark Patterns that the system supports. For each Dark Pattern, we will assign it to a category (matching those by Gray et al. [27] that we discussed in the Dark Patterns section), give the detection type, describe the impact it has on the user and the criteria used to detect the presence of this Dark Pattern.

3.3.5.1 Dark Pattern 1: Only the Opt-in option is present on the initial Cookie Dialog

Category: Forced Action **Detection Type:** Automatic

Impact on the user: This results in the user having no choice but to accept all cookies.

Criteria: Satisfies all of the following:

1. There is an opt-in button present.
2. There is not a more options button present.
3. There is not an opt-out button present.

3.3.5.2 Dark Pattern 2: The background colour of the Opt-in button leads to it being highlighted more compared to the Opt-out button.

Category: Interface Interference **Detection Type:** Automatic

Impact on the user: This makes the opt-in button stand out more to the user and means they are more likely to click the opt-in button without considering other options.

Criteria: To detect if a clickable is 'light' or 'dark' we can calculate the average colour of the grey-scale screenshot of the clickable. We use the grey-scale image as this allows us to determine the brightness of an image. If the grey-scale pixel is black then the colour of the pixel will be dark and hence not highlighted to the user. In contrast, if the grey-scale pixel is white then the colour of the pixel will be light and hence highlighted to the user. We can use the Python `opencv` Package [45] to convert the screenshot of the clickable into grey-scale and then calculate the mean colour. For each pixel, we will get a value between 0 (Dark) and 255 (Light). During testing, it was determined that considering any average value over 170 be 'light' and otherwise 'dark' was a good threshold. However, further work is needed to analyse the effectiveness of this threshold. Thus, instances of this Dark Pattern satisfy all of the following:

1. There is an opt-in option present.
2. There is an opt-out option present or a more options button present.
3. The opt-in button grey-scale colour is dark.
4. The opt-out button grey-scale colour is light and the more options button grey-scale colour is light.

3.3.5.3 Dark Pattern 3: Dialog Obstructs the window

Category: Obstruction **Detection Type:** Automatic

Impact on the user: This may hinder or entirely prevent the user from being able to interact with the rest of the web page without first interacting with the cookie dialog.

Criteria: The area (length $\times$ width) of the dialog is greater than 60% of the area of the web page. The measured screen pixels taken up by the web page will vary depending on the resolution of the monitor the system is being run on (in our case this was 1080x1920).

3.3.5.4 Dark Pattern 4: Text Content of the dialog is difficult to read

Category: Interface Interference **Detection Type:** Automatic

Impact on the user: text content of the dialog is difficult to understand to an extent that the average user may struggle to comprehend it.

Criteria: The **Flesch-Kincaid (FK)** reading ease test is a metric used to evaluate the readability of a passage of text. The FK test was originally developed for the US

Navy [30] to assess the complexity of their technical manuals. It has been widely used as a metric for automatically calculating readability, including in programs such as Microsoft word [35]. FK scores are numeric values between 0 and 100, where a higher score indicates simpler text and a lower score indicates more complex text. The FK score for a passage of text can be calculated using the following equation [30] :

$$FK = 206.835 - 1.015\left(\frac{\text{Total\_words}}{\text{Total\_sentences}}\right) - 84.6\left(\frac{\text{Total\_syllables}}{\text{Total\_words}}\right)$$

The process of accurately determining the number of syllables in a word automatically is a challenging task so we use the Python `textstat` [2] module to calculate FK scores in the system. The FK score defines any passage of text with a score of 50 or less as being difficult to read and requiring a college-level education to be understood. Thus, we define the text of any cookie dialog which has an FK score of 50 or less as being difficult to read.

3.3.5.5 Dark Pattern 5: Multiple layers to a cookie dialog

Category: Obstruction **Detection Type:** Automatic

Impact on the user: The user is required to navigate through multiple layers of a cookie dialog to select their preferences. This may push users to accept all cookies as it requires substantially less effort.

Criteria: a more options button is present on the valid dialog.

3.3.5.6 Dark Pattern 6: Ambiguous Close button is present on the dialog

Category: Interface Interference **Detection Type:** Automatic

Impact on the user: The purpose of this button may be ambiguous to the user. They may not know if clicking this button will opt-in, opt-out or make another cookie setting choice on their behalf.

Criteria: a close option button is present on the valid dialog.

3.3.5.7 Dark Pattern 7: Multiple distinct Cookie Dialogs present on a page

Category: Interface Interference **Detection Type:** Automatic

Impact on the user: The user may be uncertain about which cookie dialog they should interact with and different dialogs may have different options or instructions available.

Criteria: There is more than 1 distinct candidate dialogs which means the following statements hold:

1. All candidate dialogs being considered all have a score greater than 0.
2. The HTML content of at least one candidate is not the same as or a sub-string of another candidate's HTML content.

3.3.5.8 Dark Pattern 8: At least one Preference Slider is enabled by default.

Category: Sneaking **Detection Type:** Automatic

Impact on the user: There can be many preference sliders on a dialog so one or more being enabled by default can be easily missed by the user. This also adds additional effort for the user to reject all non-essential cookies.

Criteria: Selenium provides two useful functions to detect if a preference slider is enabled or not. The `is_selected()` [9] function can be used to check if a preference slider has been selected, this allows us to check if a preference slider has been automatically enabled by the website without the user's consent. The `is_enabled()` method [9] allows us to check if an element can be interacted with by the user, this allows us to tell if the user can disable this preference slider or not. Commonly, many dialogs have a preference slider for 'necessary cookies' which cannot be interacted with and is enabled by default. If both of these functions return `True` then this means the Preference Slider has been enabled by default and can be interacted with by the user. Thus, the criteria for this Dark Pattern is that a preference slider is present and it satisfies the following:

1. The preference slider is enabled (`is_enabled` function returns True).
2. The preference slider is selected (`is_selected` function returns True).

3.3.5.9 Dark Pattern 9: Clicking the Close button leads to more cookies being selected.

Category: Sneaking **Detection Type:** Automatic

Impact on the user: Without informing the user that clicking the close button will set more cookies the dialog has selected more cookies without asking for informed consent from the user.

Criteria: We use id-like cookies as they are more likely to be tracking cookies. In many cases, it was observed that the website would set a cookie to indicate the user's cookie preference or for other reasons such as language. By just comparing id-like cookies we hope to have a good way to detect if there is an increase in tracking cookies. However, not all id-like cookies will be tracking cookies and there is no way to tell the exact purpose of cookies for a website you did not develop yourself. In a study by Sanchez-Rola et al. [20] they distinguish id-like cookies using the `zxcvbn` password strength algorithm. This allows us to assess the uniqueness of a string thus allowing us to gauge how suitable it is to be a unique identifier. We use this approach to determine the strength of the cookie value attribute in being a unique identifier. The value attribute of a cookie can contain multiple distinct sub-strings. To retrieve these sub-strings we split the string using following the delimiters:

```
: , = , . , % , |
```

Additionally to allow tracking across websites, id-like cookies must also be persistent, that is they must not deleted immediately after the browsing session is closed. In their study Englehardt et al. [15] consider cookies with an expiry date of over 90 days to be identifier cookies. Typically, tracking id-like cookies have a long expiration date set to allow a more complete user profile to be constructed. To determine if a cookie is id-like it must meet the following criteria:

1. The cookie is set to expire more than 90 days after it was set.
2. The `zxcvbn` [20] score of any component of the value attribute of the cookie is greater than 3 (this is equivalent to 100 million guesses).

Therefore the criteria for this Dark Pattern is:

1. An initial set of cookies was collected when the website was first opened.
2. There is a close button present on the cookie dialog.
3. A set of cookies was collected after the close button was interacted with.
4. The number of id-like cookies in the set of cookies collected after the close button was clicked is greater than the number of id-like cookies in the initial set of cookies.

3.3.5.10 Dark Pattern 10: Takes more clicks to Opt-out than Opt-in or Opt-out option is not visible.

Category: Obstruction **Detection Type:** Manual

Impact on the user: Additional effort is required by the user to opt-out than to opt-in which may make sway some users to simply opt-in rather than undergo the additional effort.

Criteria: The number of clicks it took the user to opt-out is more than the number it took them to opt-in during the manual interaction with the cookie dialog.

3.3.5.11 Dark Pattern 11: More cookies are set regardless of the Opt-out button being clicked

Category: Sneaking **Detection Type:** Automatic

Impact on the user: Despite the user opting out to reject all non-essential cookies the dialog has disregarded their choice and selected more cookies regardless.

Criteria: We take the same approach as we described for Dark Pattern 9 to detect id-like cookies. Our criteria for this Dark Pattern is, there are more id-like cookies stored after the opt-out option is clicked compared to before which means:

1. An initial set of cookies was collected when the website was first opened.
2. There is an opt-out option present on the cookie dialog.
3. A set of cookies was collected after the opt-out option was interacted with.
4. The number of id-like cookies stored in the set of cookies collected after the opt-out option was clicked is greater than the number of id-like cookies in the initial set of cookies.

3.3.5.12 Dark Pattern 12: Poorly Labelled preference sliders

Category: Sneaking **Detection Type:** Manual

Impact on the user: The user may be unsure about what one or more preference sliders do when enabled which may lead to them unintentionally opting in for more cookies that they wish to.

Criteria: This is decided by the user during manual input of Dark Patterns. General guidance is that the user believes that labels for one or more preference sliders are unclear to the extent that their purpose is ambiguous.

3.3.5.13 Dark Pattern 13: In the context of the Cookie Dialog text the standard meaning of the Opt-in and Opt-out buttons is inverted.

Category: Interface Interference **Detection Type:** Manual

Impact on the user: The typical layout of a cookie dialog has the opt-in button worded affirmatively and the opt-out button worded negatively. Switching this standard ordering may lead to the user unintentionally selecting more cookies than they wished to.

Criteria: This is decided by the user during manual input of Dark Patterns. General guidance is that the user believes that the context of the opt-in button is worded negatively and the opt-out button worded affirmatively to the extent that it may confuse other users.

3.3.5.14 Dark Pattern 14: Opt-out button is named to guilt the user for selecting it

Category: Interface Interference **Detection Type:** Manual

Impact on the user: The user may feel guilty or that they are missing out on a feature by opting out of non-essential cookies.

Criteria: This is decided by the user during manual input of Dark Patterns. General guidance is that the user believes that they or other users would feel guilty for selecting the opt-out button.

3.3.5.15 Dark Pattern 15: When the user clicks the Opt-out button they are asked to confirm their choice

Category: Nagging **Detection Type:** Manual

Impact on the user: The user has to undergo additional effort when opting out of non-essential cookies.

Criteria: The opt-out button should be followed by an additional action to confirm the user's choice and the opt-in button should not have this additional action.

Chapter 4

Evaluation

To evaluate the accuracy of our system manual validation was conducted on two sets of websites and then a larger data set was used to automatically collect results.

Top 500 data set: the first set used to evaluate the system was the top 500 websites of 2021 from the Tranco list. This was chosen to allow us to assess the cookie dialogs on popular websites. This data set was collected in Manual mode of the tool between 22/01/22 and 29/01/22 and all data were manually validated.

Random 500 data set: the second data set used to evaluate the system was a random subset of 500 websites from the same Tranco list. This was chosen to avoid bias towards the top 500 websites. This data set was collected in Manual mode of the tool on 14/02/22 and all data were manually validated.

Random 10k data set: the final data set was collected using another random subset of 10,000 websites from the same Tranco list. This was chosen to allow us to assess a larger collection of websites to gain a better idea of what Dark Patterns are present across the web. This data set will overlap with the previous two data sets and was collected in Automatic Mode between 15/02/22 and 24/02/22. This data set was not manually validated so we cannot assess the accuracy of this data set.

The full details of each data set, including the full lists of domains used, can be found in the appendix.

4.1 Dialog Collection

In the Top 500 data set there were 235 websites with a cookie dialog present, the system was successful in finding the correct dialog for 233 websites, giving us an recall[1] of 99.5 %. Of the 237 positive results by the system, only 4 of these were false positives, giving us a precision [2] of 98.3 %. Overall for the Top 500 websites the system had a cookie dialog collection accuracy[3] of 98.8 %.

[1]Recall = $\frac{\text{TP}}{\text{TP+FN}}$
[2]Precision = $\frac{\text{TP}}{\text{TP+TN}}$
[3]Accuracy = $\frac{\text{TP+TN}}{\text{TP+TN+FP+FN}}$

An interesting false positive case that was encountered during the collection of this data set is shown in Figure 4.1. This website contains a dialog which at first glance appears to be a cookie dialog. However, upon further inspection, this dialog and its attached privacy policy do not mention cookies so this is not a cookie dialog.

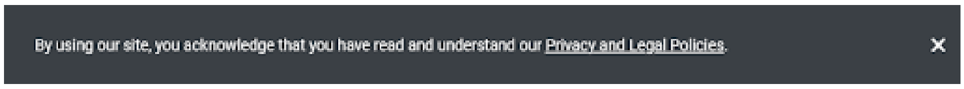

By using our site, you acknowledge that you have read and understand our Privacy and Legal Policies.

Figure 4.1: An example of website dialog which is not a cookie dialog (Source: iden-trust.com)

		Actual	
		Positive	Negative
Expected	Positive	233	4
	Negative	2	261

Table 4.1: Dialog detection confusion matrix for the Top 500 websites.

In the Random 500 websites data set there were 98 websites with a cookie dialog present, the system was successful in finding the correct dialog for 97 websites, giving us a recall of 99.0 %. Of the 102 positive results by the system, only 5 of these were false positives, giving us a precision of 95.1 %. Overall for the Top 500 websites, the system had a cookie dialog collection accuracy of 98.8 %.

An interesting false negative edge case was encountered on the website kaffee-partner.de during the collection of the Random 500 websites. This website does have a cookie dialog present but it is wrapped inside a shadow DOM. The shadow DOM allows a web page to attach a hidden separated DOM to an element [41]. The shadow DOM is not supported by Selenium so there is no way for our current system to handle this case. This is the only instance of this edge case that was found during manual validation out of 1000 websites.

		Actual	
		Positive	Negative
Expected	Positive	97	5
	Negative	1	397

Table 4.2: Dialog detection confusion matrix for the Random 500 websites.

It should be noted that due to the fast-changing nature of website development and the ability of CSS Selectors Lists to keep themselves updated will mean the accuracy varies depending on the websites chosen and the date they were analysed.

Since we used the same Tranco list for all three data sets, the websites in Random 10k data set overlaps with our two smaller data sets. The Random 10k data set was collected around about the same date as our smaller data sets. For these reasons we can expect our this data set to have similar accuracy to the two smaller data sets, giving us an accuracy of roughly 98.8% dialog collection accuracy.

This is an improvement on the previous study by Petronyte [47] who reported an accuracy of 93.0% using the CSS selectors and keyword filtering only. This suggests that our ranking based system (described in Ranking all the Candidate Dialogs section) was successful in improving cookie dialog collection accuracy.

The average time our system took to analyse a website across all 3 data sets was 46.3 seconds. This is well within the 30-minute requirement specified by our non-functional requirement for timeliness.

4.2 Clickable Location & Classification

In the Top 500 data set, 1211 of the 1273 total clickables were correctly classified, meaning the system had a 95.1% clickable classification accuracy.

In the Random 500 data set, 639 of the 656 total clickables were correctly classified, meaning the system had a 97.4% clickable classification accuracy.

Using the same logic as with dialog collection, we can surmise that the larger data set will have an accuracy of roughly 95.1-97.4%.

Data Set	**Count**	**Opt-in**	**Opt-out**	**More Options**	**Preference Slider**	**Confirm Preference**	**Close**	**Policy Link**	**Other**	**Duplicate**
Top 500	Actual	242	57	232	45	10	44	281	231	131
	Expected	232	60	211	37	9	39	283	262	140
Random 500	Actual	102	25	39	56	9	15	94	82	234
	Expected	100	22	35	56	9	16	94	84	240

Table 4.3: Actual and expected values for each clickable type.

4.3 Dark Pattern Detection

In the Top 500 data set, 810 of 814 total Dark Patterns were detected and there were 3 false positives. Meaning the system had a 99.0% Dark Pattern detection accuracy.

In the Random 500 data set, 558 of 561 total Dark Patterns were detected and there was 1 false positive. Meaning the system had a 99.2% Dark Pattern detection accuracy.

Interestingly, in both data sets, all the false positives appeared for Dark Pattern 7 (Multiple distinct Cookie Dialogs present on a page), which may suggest that the criteria for detecting duplicate dialogs was too relaxed. In future, more stringent criteria could be applied for this Dark Pattern. For example, the size and position of a candidate dialogs could also be compared to help reduce the false positive rate.

Data Set	**Count**	**DP1**	**DP2**	**DP3**	**DP4**	**DP5**	**DP6**	**DP7**	**DP8**	**DP9**	**DP11**
Top 500	Actual	31	75	55	110	172	34	11	1	5	80
	Expected	32	77	55	110	172	34	8	1	5	80
Random 500	Actual	36	18	17	45	30	11	11	1	1	21
	Expected	36	20	17	45	30	11	10	1	1	21

Table 4.4: Actual and expected values for each Dark Pattern.

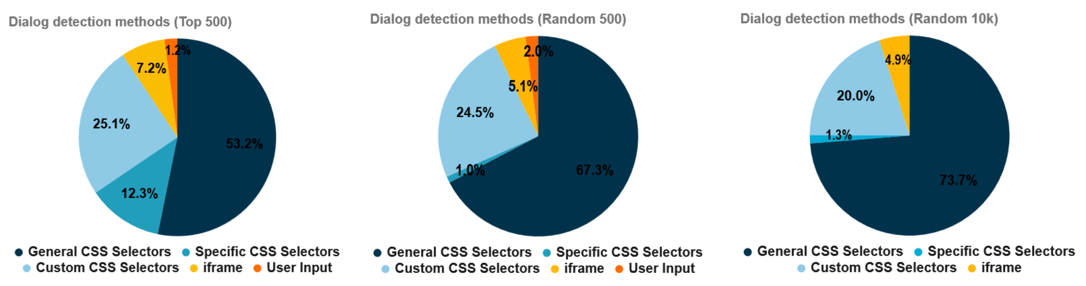
Chapter 5

Results

5.1 Cookie Dialogs

In the Top 500 data set, there were a total of 235 websites that displayed cookie dialogs, meaning that 47% of websites displayed a cookie dialog. In the Random 500 data set, there were a total of 98 websites that displayed a cookie dialog, meaning that 19.6% of websites displayed a cookie dialog. In the Random 10k data set, there were a total of 2092 websites that displayed a cookie dialog, meaning that 20.92% of websites displayed a cookie dialog. These results show that websites in the Top 500 are more than 2 times as likely to display a cookie dialog than those observed in the Random 500/ Random 10k websites.

In Finding all candidate dialogs section, we described the methods used to locate all candidate dialogs. We will now explore which of these methods were the most effective in locating the valid cookie dialog. In Figure 5.1 we can see a breakdown of the methods used to locate the cookie dialogs in each data set. The 'User Input' method is the method used during manual validation by the user to specify the correct dialog if the detected one is incorrect. It should be noted that since the Random 10k data set was collected in Automatic mode that no dialogs were collected using this method.



(a) Top 500 data set (b) Random 500 data set (c) Random 10k data set

Figure 5.1: Graphs showing detection methods used to find dialogs in each data set.

In all 3 data sets, we can see that the General CSS Selectors were responsible for locating more than half of all the final dialogs. This means that the system is still heavily reliant on the ability of CSS selector lists to be kept updated. An important observation from Figure 5.1 is the relatively large percentage (12.3%) of dialogs detected using Specific CSS Selectors in the Top 500 data set and the small percentages in the Random 500 (1%)/Random 10k (1.3%) data sets. This is likely because CSS Selector lists are publicly updated. This means that the more popular websites, particularly those in the Top 500 websites, are more frequently updated in CSS Selector lists.

5.2 Clickables

We will first explore our results to see what clickable elements were commonly found on cookie dialogs. We will also comment on the effort it takes for users to opt-out of cookies. During the manual validation process, the researcher manually rejected cookies. The number of clicks and the time this took them was recorded. We can use these two metrics as an approximation of the effort a typical user would have to go through to reject cookies. Note that we can only do this for the two manually collected data sets and not the automatically-collected data set.

In the Top 500 websites, there was a total of 830 clickables detected, with an average of 3.5 clickables per dialog. It took the user an average of 3.0 clicks and an average of 7.6 seconds to reject cookies.

In the Random 500 websites, there was a total of 328 clickables detected, with an average of 3.3 clickables per dialog. It took the user an average of 2.4 clicks and an average of 6.4 seconds to reject cookies.

Comparing the manually validated Top 500 and Random 500 data sets we can see that on average it took the researcher more time and clicks to reject cookies for websites in the Top 500 data set. These results are in line with our observation for Dark Pattern 11 that more websites in the Top 500 have multiple layers for users to go through to opt-out from cookies.

In the Random 10k websites, there were a total of 7337 clickables detected, with an average of 3.5 clickables per dialog.

Figure 5.2 shows a breakdown of the percentage of websites with at least one occurrence of each clickable type. For all data sets, we can see the opt-in option appeared on a higher percentage of websites compared to the opt-out button. This suggests that many websites choose to make it easier for a user to opt-in to all cookies and harder for them to opt-out from cookies.

A major difference between the data sets can be observed for the 'more options' clickable. 73.1 % of cookie dialogs in the Top 500 websites have a 'more options' button present whereas just 30.6% and 30.1% of websites in the Random 500 and Random 10k data sets. Since there is no significant increase in the opt-out button for the Random 500 and Random 10k data sets it is likely that many cookie dialogs have no way for users to opt-out.

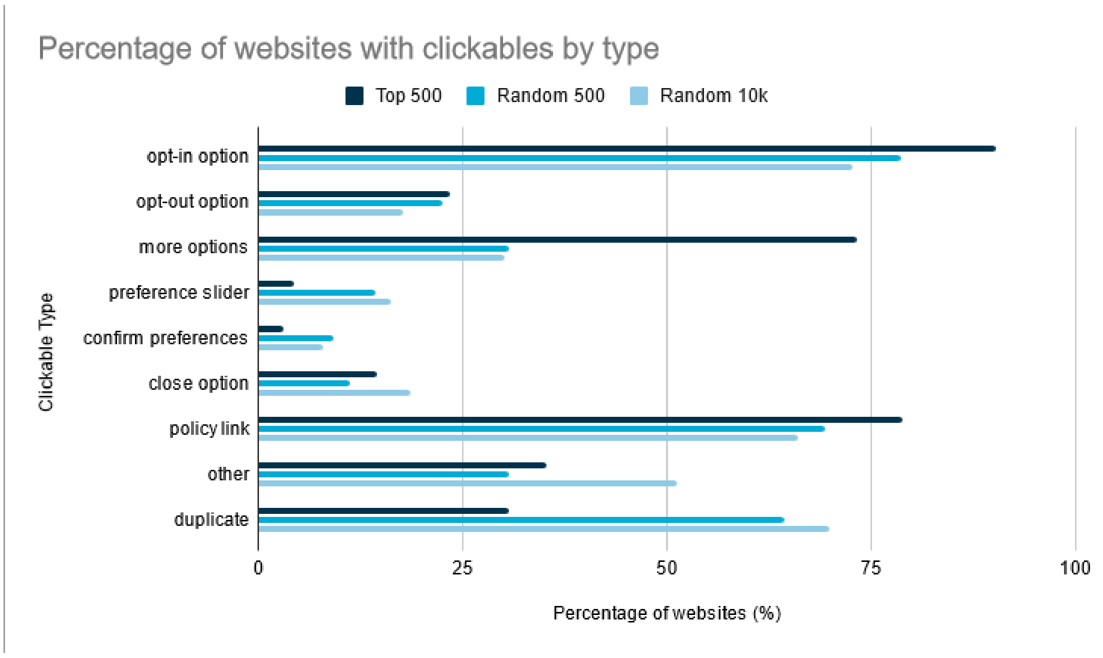


Figure 5.2: Graph showing percentage of websites with at least one occurrence of each clickable type.

5.3 Cookie Setting Behaviour

Figure 5.3 shows that the vast majority of websites set at least one cookie before getting consent from the user. That is 94% of websites in the Top 500 data set, 96.9% in the Random 500 data set and 95.1% in the Random 10k data set.

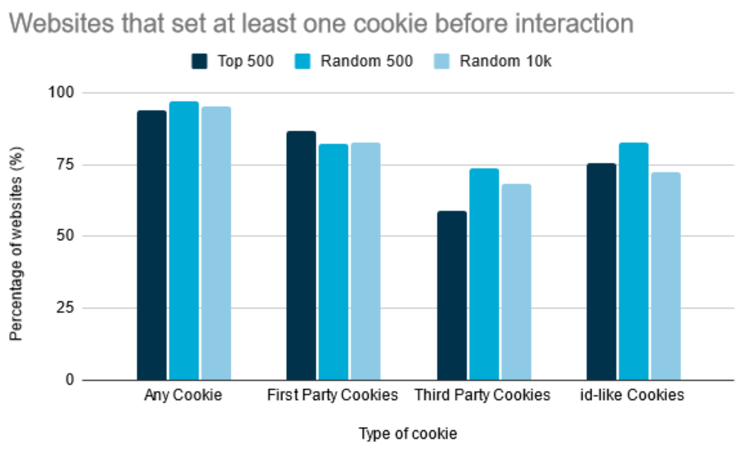


Figure 5.3: Graph showing the percentage of websites that set at least one cookie before any interaction from the user.

The cookie setting behaviour observed for each data set is shown in Figure 5.4 for the Top 500 data set, Figure 5.5 for the Random 500 data set and Figure 5.6 for the Random

10k data set. These figures show a comparison of the average number of cookies upon initially loading the web page and after interaction with various types of clickable.

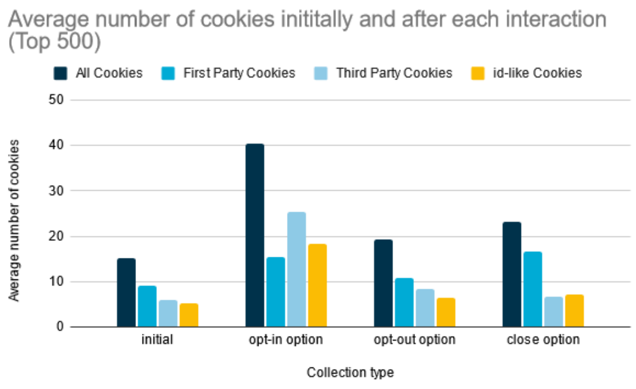


Figure 5.4: The average number of cookies after each interaction for the Top 500 data set.

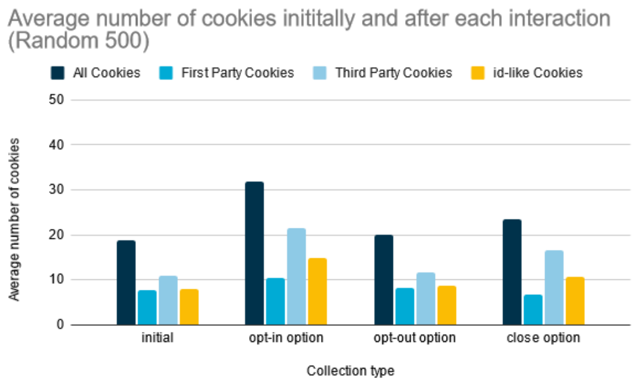


Figure 5.5: The average number of cookies after each interaction for the Random 500 data set.

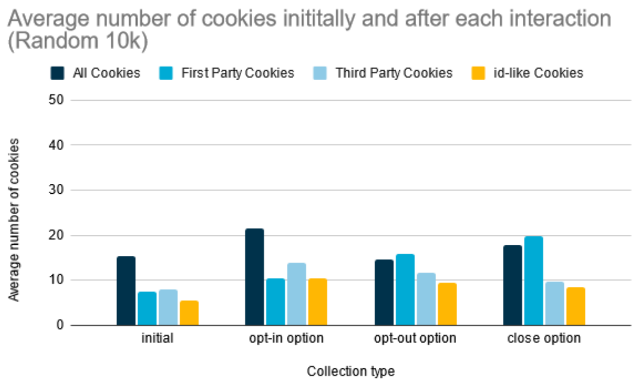


Figure 5.6: The average number of cookies after each interaction for the Random 10k data set.

From Figure 5.4, Figure 5.5 and Figure 5.6 we can make several interesting conclusions. Firstly, we observe that for all data sets the average number of id-like cookies increases after the opt-in option is clicked. This is expected as users are giving their consent for websites to enable more cookies.

Secondly, we observe that for all data sets the average number of cookies increases after the opt-out option is clicked. A slight increase in first-party cookies is expected as these will likely be the necessary cookies that websites are allowed to set. However, we also see an increase in third-party and id-like cookies. These are typically used for non-essential purposes such as tracking. It is concerning to see that many websites choose to set more of these types of cookies.

Finally, we observe that for all data sets that the average number of id-like cookies decreases after the close button is clicked. The purpose of the close button is often ambiguous to users. However, this cookie setting behaviour may suggest that the purpose of the close button is often similar to that of the opt-out button.

5.4 Dark Patterns

Table 5.1 shows the total number of Dark Patterns found and their prevalence across each data set. The prevalence is simply the percentage of cookie dialogs that contained this Dark Pattern. Note that Dark Patterns 10 and 12-15 are not applicable for Random 10k data set since they require manual input and this data set was collected in automatic mode.

From the 235 dialogs collected in the Top 500 data set, 227 dialogs or 96.6% of them had at least one Dark pattern present. From the 98 dialogs collected in the Random 500 data set, 94 dialogs or 95.9% of them had at least one Dark pattern present. We can compare the two manually collected data sets as they have the same number of Dark Patterns collected for them. We can see that there were more Dark Patterns located in the Top 500 websites than the Random 500.

We cannot compare the Random 10k data set to the other data sets as it only contains automatically detected Dark Patterns. From the 2092 dialogs collected in the Random 10k data set, 1872 dialogs or 89.5% of them had at least one Dark pattern present.

Data Set	# of Dialogs	Count	DP1	DP2	DP3	DP4	DP5	DP6	DP7	DP8	DP9	DP10	DP11	DP12	DP13	DP14	DP15
Top 500	**235**	**Total**	32	77	55	110	172	34	8	1	5	178	80	13	0	2	1
		Prevalence	13.6%	32.8%	23.4%	46.8%	73.1%	14%	3.4%	0.4%	2.1%	75.7%	34.0%	5.5%	0.0%	0.9%	0.4%
Random 500	**98**	**Total**	36	20	17	45	30	11	10	1	1	80	21	2	0	1	1
		Prevalence	36.7%	20.4%	17.3%	45.9%	30.6%	11.2%	10.2%	1.0%	1.0%	81.6%	21.4%	2.0%	0.0%	1.0%	1.0%
Random 10k	**2092**	**Total**	751	323	320	1006	619	379	169	55	59	-	93	-	-	-	-
		Prevalence	35.9%	15.4%	15.3%	48.6%	29.6%	18.1%	8.1%	2.6%	2.8%	-	4.3%	-	-	-	-

Table 5.1: The total number and prevalence of Dark Patterns in each data set.

In the following paragraphs, we will individually analyse the results for each automatically detected Dark Pattern. To allow us to compare the prevalence across all data sets will do this for the automatically detected Dark Patterns only. So we will omit the manually inputted Dark Patterns 10, 12, 13, 14 and 15.

Dark Pattern 1 is 'Only Opt-in option is present on initial Cookie Dialog'. This Dark Pattern has a prevalence of 36.7% for Random 500 and 35.9% for Random 10k which is much higher than the prevalence score of 13.6% for Top 500. This suggests that more popular websites present more than just the basic opt-in option which is likely due to the fact they will have larger design teams and possibly even legal teams which will have helped them ensure their cookie dialog is compliant with relevant legislation.

Dark Pattern 2 is 'Background colour of the Opt-in button leads to it being highlighted more compared to the Opt-out button. This Dark Pattern has a prevalence of 32.8% for the Top 500, 20.4% for the Random 500 and 15.4% for the Random 10k. It is interesting to see that websites in the Top 500 are more likely to have this Dark Pattern. A study by Fernandez et al. [11] found that having a highlighted opt-in button had a significant effect on user interactions. By highlighting the Opt-in button it is likely these websites hope to subtly draw users towards accepting more cookies.

Dark Pattern 3 is 'Dialog Obstructs the window'. This Dark Pattern has a prevalence of 23.4% for the Top 500, 17.3% for the Random 500 and 15.3% for the Random 10k. These results suggest that the majority of websites do not place their cookie dialogs in a position that stops users from interacting with the web page. This ties in with the results in the study by Soe et al. [21] where during a manual collection they observed the position of cookie dialogs was rarely seen to block users.

Dark Pattern 4 is 'Text Content of the dialog is difficult to read'. This Dark Pattern has a prevalence of 46.8% for the Top 500, 45.9% for the Random 500 and 48.6% for the Random 10k. Meaning these results are fairly similar across data sets. We described the FK score in the Dark Patterns section and noted that dialogs with an FK score of 50 or less as being difficult to read and requiring a college-level education to be understood. The average FK scores for each data set were as follows: 47.2 for the Top 500 data set, 49.2 for the Random 500 data set and 45.9 from the Random 10k data set. It is concerning to see that nearly half of all cookie dialogs are considered difficult for the average user to read.

Dark Pattern 5 is 'Multiple layers to a cookie dialog'. This Dark Pattern has a prevalence of 73.1% for the Top 500, 30.6% for the Random 500 and 29.6% for the Random 10k. This ties in with our observation in the Clickables Results section that the Top 500 data set contains the 'more options' clickable much more than the Random 500 and Random 10k data sets.

Dark Pattern 6 is the 'Ambiguous Close button is present on the dialog'. This Dark Pattern has a prevalence of 14% for the Top 500, 11.2% for the Random 500 and 18.1% for the Random 10k. The majority of websites do not have this clickable which is good as its purpose is often ambiguous. We will discuss the observed behaviour of this button in the Dark Pattern 9 discussion.

Dark Pattern 7 is 'Multiple distinct Cookie Dialogs present on a page'. This Dark Pattern has a prevalence of 3.4% for the Top 500, 10.2% for the Random 500 and 8.1% for the Random 10k. Meaning this Dark pattern appears much less in the Top 500 data set. The presence of multiple cookie dialogs is likely a mistake by the website designer, as it is very confusing to users. It is likely on popular websites, such as those in the Top

500, that this issue would be spotted very quickly and removed.

Dark Pattern 8 is ’At least one Preference Slider is enabled by default’. To truly assess the prevalence of this Dark Pattern we must compare it to the number of preference sliders that were present. For the Top 500, there were 9 dialogs with preference sliders present and just 1 of them or 11.1% had one of these enabled by default. It was the same case for the Random 500 and Random 10k, where 7.7% and 21.5% of preference sliders respectively had a cookie setting behaviour which set more id-like cookies. Overall, for all data sets the vast majority of cookie dialogs with preference sliders do not have them enabled by default.

Dark Pattern 9 is ’Clicking the Close button leads to more cookies being selected’. To truly assess the prevalence of this Dark Pattern we must compare it to the number of close buttons which were present. For the Top 500, there were 34 dialogs with close buttons present and 5 of them or 14.7% had a cookie setting behaviour that set more id-like cookies. It was the same case for the Random 500 and Random 10k where 9.1% and 15.2% of close buttons respectively had a cookie setting behaviour which set more id-like cookies. Overall, for all data sets the vast majority of close buttons do not set more cookies. This suggests that for many websites the ambiguous behaviour of the close button is to opt-out which is good from a privacy perspective.

Dark Pattern 11 is ’More cookies are set regardless of Opt-out button being clicked’. To truly assess the prevalence of this Dark Pattern we must compare it to the number of cookie dialogs where there was an option to opt-out. Note that for Top 500 and Random 500 data sets the user manually rejected cookies so there will be more occurrences than there are opt-out buttons in these data sets. For the automatically collected Random 10k data set cookies were only collected if an opt-out button is present. For the Top 500 data set, there were 232 dialogs with a way to opt-out and 80 of them or 34.5% set more id-like cookies after the user rejected consent. For the Random 500 data set, there were 98 dialogs with a way to opt-out and 21 of them or 21.4% set more id-like cookies after the user rejected consent. For the Random 10k data set there were 368 dialogs with an opt-out button present and 93 of them or 25.2% set more id-like cookies after the user rejected consent. It is concerning to see that the Top 500 websites have the highest prevalence of Dark Patterns among our data sets. This means that around one-third of the Top 500 websites ignore the user’s request to refuse any more non-essential cookies which is a damning indictment.

Figure 5.7 provides a breakdown of the total number of Dark Patterns detected on individual websites. Overall, the Top 500 data set had an average of 3.2 Dark Patterns on each cookie dialog, the Random 500 data set had an average of 2.8 and the Random 10k data set had an average of 1.8. It is important to note that the Random 10k data set did not include the manually assessed Dark Patterns which is likely why fewer Dark Patterns were detected. It is interesting to see that the more popular websites in the Top 500 have a higher number of Dark Patterns which may suggest that they have been designed specifically to include more Dark Patterns.

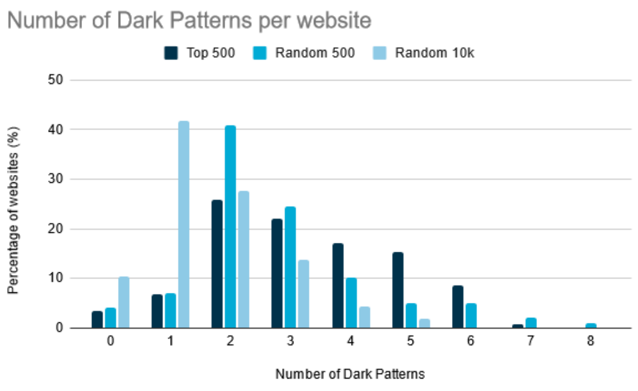


Figure 5.7: The number of Dark Patterns per website across all data sets.

To assess if particular categories of websites have more Dark Patterns than others we used the Web Filtering service by FortiGuard [33]. This service is used to provide a registry for malicious domains but also gives the category of web pages hosted on the domain. For example, the domain 'google.com' falls under the 'Search Engines and Portals' category. We used this service to assign all the websites in our 3 data sets to a category.

Table 5.2 shows the Top 10 categories of websites with the most Dark Patterns on websites. The only category common in the Top 10's of all 3 data sets is 'Personal Websites and Blogs'. Across all data sets, we analysed websites from 82 different categories which demonstrates that we have covered a wide range of websites. However, some of the categories provided by FortiGuard were quite niche and only a few instances of these categories were found within our data sets. To truly assess if certain categories of websites had more Dark Patterns than others we would need to to have a similar number of websites in each category. We could achieve this by collecting a larger data set or selecting an equal number of websites in each category.

Top 500

Category	**Average # of Dark Patterns**
Internet Radio and TV	6
Search Engines and Portals	5.2
Remote Access	5
Travel	5
Instant Messaging	4.3
Finance and Banking	4.2
Social Networking	3.7
Personal Websites and Blogs	3.7
Auction	3.5
Health and Wellness	3.5

Random 500

Category	**Average # of Dark Patterns**
Society and Lifestyles	8
Internet Radio and TV	6
Personal Websites and Blogs	6
Advertising	5
Other Adult Materials	5
Shopping	3.5
Finance and Banking	3
General Organizations	3
Global Religion	3
Government and Legal Organizations	3

Random 10k

Category	**Average # of Dark Patterns**
Auction	3
File Sharing and Storage	2.7
Web	2.7
Child Education	2.5
Information and Computer Security	2.5
Real Estate	2.5
Malicious Websites	2.3
Personal Websites and Blogs	2.1
Travel	2.1
Restaurant and Dining	2.1

Table 5.2: Top 10 Categories of websites with the most Dark Patterns in each data set.

Chapter 6

Conclusion

Overall, we have achieved our main goal of designing and implementing a system that can automatically detect Dark Patterns on a cookie dialog. To enable this our system can collect cookie dialogs, locate their clickables and analyse the cookie setting behaviour of the dialog.

We have improved the accuracy of collecting cookie dialogs over previous studies [36][47] using a new ranking-based approach. We could further improve the dialog collection accuracy of our system by re-generating the N-grams from a larger set of clickables, such as those collected in the Random 10k data set. This process could be further improved by expanding the number of factors included in the ranking process. One possible additional factor is the size of dialog which may help to reduce false positives. Another factor that may improve accuracy would be to locate clickable elements on each candidate dialog, although this would certainly be extremely time-consuming. Finally, the weights assigned to each factor could be more finely tuned through further experimentation.

We have also extended the set of clickables that our system can locate and classify to include preference sliders. The results of this project show that the majority of cookie dialogs with preference sliders do not have them enabled by default. We observed that in the majority of cases the behaviour of the opt-in and opt-out buttons was as expected. We also observed that interaction with the ambiguous close button often led to less id-like cookies being set. This suggests that the purpose of the close button is often similar to that of the opt-out button. Further improvements to the clickable classification accuracy of our system could be made by re-generating the list of keywords from a larger set of clickables, such as the set clickables we collected in the Random 10k data set.

A limitation of the system is that we cannot automatically opt-out from cookies if it involves us having to navigate through several layers of options. This is due to complexity and variation in the layout of cookie dialogs which means there is no trivial way to opt-out from all cookies. Ideally, we would want to find a way to automate this process as it would allow us to fully assess the cookie setting behaviour of cookie dialogs automatically. However, we could not identify any possible way for this to be achieved as in some cases human judgement is required to opt-out from cookies.

Our system can detect a total of 10 Dark Patterns automatically with an accuracy of 99.0-99.2%. A further 5 Dark Patterns can be inputted manually. Ideally, we would want to detect all of these Dark Patterns automatically to allow them to be assessed on a large number of websites. However, this was not possible within the timescale of this project. From a sample of 10,000 websites our study has shown that 89.5% of cookie dialogs have at least one Dark Pattern present.

The resulting system has allowed us to analyse Dark Patterns on over 10,000 websites, a scale that would have been very difficult without our automated tool. This has enabled us to collect over 2000 cookie dialogs which we intend to make publicly available for usage in future research. To give us an even better understanding of the prevalence of Dark Patterns across the web it would be interesting to scale up the system to allow millions of websites to be analysed. The current bottleneck in the system is waiting for cookies to be set. To resolve this bottleneck we would have to run multiple concurrent instances of the system, with each instance analysing a different domain at once. To provide enough compute power and network bandwidth for scaling up the system we would need to use a cloud computing provider. The system could work well if deployed using containerisation technologies such as Docker [8] which would allow us to truly ensure there was no cross-contamination between instances and scale the system almost infinitely.

By defining the types of Dark Patterns commonly found on cookie dialogs, this research has shown the common design techniques used by web designers to nudge users towards accepting more cookies. It is hoped that the results of this study will help future research into the better enforcement of laws such as the PECR and GDPR [43]. Furthermore, by pointing out and providing examples of Dark Patterns we hope this study will help users to be aware of Dark Patterns and make more informed consent choices.

This results of this project have shown the fluctuation in the number of Dark Patterns between websites. This highlights the huge variance in cookie dialog designs across the web. The internet is currently in a situation where the variance in cookie dialogs designs has made user interaction with cookie dialogs difficult. This is because users have to carefully consider their cookie consent choices separately at each website which can be detrimental to user experience. An ideal scenario would allow users to specify their privacy settings for all websites rather than just for individual websites. Several technologies such as the Platform for Privacy Preferences Project (P3P) [53] and the Transparency & Consent Framework (TCF) [22] have attempted to provide a single way for users to specify their consent preferences. P3P is no longer supported by any web browsers [34] and was deemed far to complex for users [49]. Although TCF has been adopted by many EU websites, it was recently ruled to not comply with EU law [23]. Until a technology exists which provides compliance with the relevant legislation and is easily accessible for users, we will have to contend with cookie dialogs and the Dark Patterns present on them.

Bibliography

[1] Alexa. The top 500 sites on the web. https://www.alexa.com/topsites/, 2021. Last accessed 17 October 2021.

[2] Shivam Bansal and Chaitanya Aggarwal. pypi textstat. https://pypi.org/project/textstat/, 2022. Last accessed 27 March 2022.

[3] Harry Brignull. Dark patterns: dirty tricks designers use to make people do stuff. https://90percentofeverything.com/2010/07/08/dark-patterns-dirty-tricks-designers-use-to-make-people-do-stuff/, 2010. Last accessed 16 October 2021.

[4] Harry Brignull. Twitter: Dark pattern. https://twitter.com/darkpatterns, 2010. Last accessed 19 October 2021.

[5] Harry Brignull. Types of dark pattern. https://www.darkpatterns.org/types-of-dark-pattern, 2010. Last accessed 17 October 2021.

[6] Matt Chaput. pypi stemming 1.0. https://pypi.org/project/stemming/1.0/, 2010. Last accessed 27 March 2022.

[7] collin M. Barrett. Filterlists. https://filterlists.com/, 2022. Last accessed 30 March 2022.

[8] Docker. Docker inc. https://www.docker.com/, 2022. Last accessed 27 March 2022.

[9] Selenium Docs. Interacting with web elements. https://www.selenium.dev/documentation/webdriver/elements/interactions/#click, 2022. Last accessed 27 March 2022.

[10] Selenium Docs. Working with iframes and frames. https://www.selenium.dev/documentation/webdriver/browser/frames/, 2022. Last accessed 27 March 2022.

[11] Bermejo Fernandez et al. ”this website uses nudging: Mturk workers’ behaviour on cookie consent notices.”. In *Proceedings of the ACM on Human-Computer Interaction 5, no. CSCW2*, pages 1–22, 2021.

[12] Christine Utz et al. (un)informed consent: Studying gdpr consent notices in the field. In *CM SIGSAC Conference on Computer and Communications Security (CCS ’19),*, https://doi.org/10.1145/3319535.3354212, 2019.

[13] Dageling et al. We value your privacy ... now take some cookies: Measuring the gdpr's impact on web privacy. In *arXiv preprint arXiv:1808.05096.*, 2018.

[14] Eijk et al. The impact of user location on cookie notices (inside and outside of the european union). In *Workshop on Technology and Consumer Protection (ConPro'19) IEEE*, 2019.

[15] Englehardt et al. Online tracking: A 1-million-site measurement and analysis. In *Proceedings of the 2016 ACM SIGSAC confer- ence on computer and communications security.*, page 1388–1401, 2016.

[16] Krisham et al. Dark patterns in the wild: Review of cookie disclaimer designs on top 500 german websites. In *European Symposium on Usable Security 2021 (EuroUSEC '21),*, page 8, https://doi.org/10.1145/3481357.3481516, 2021.

[17] Machuletz et al. Multiple purposes, multiple problems: A user study of consent dialogs after gdpr. In *arXiv preprint arXiv:1908.10048.*, 2019.

[18] Matte et al. Do cookie banners respect my choice? measuring legal compliance of banners from iab europe's transparency and consent framework. In *In2020 IEEE Symposium on Security and Privacy (SP) 2020*, pages 791–809, 2020.

[19] Pochat et al. Tranco a research-oriented top sites ranking hardened against manipulation. https://tranco-list.eu/, 2021. Last accessed 19 October 2021.

[20] Sanchez-Rola et al. "can i opt out yet? gdpr and the global illusion of cookie control.". In *In Proceedings of the 2019 ACM Asia conference on computer and communications security*, pages 340–351, 2019.

[21] Soe et al. "circumvention by design-dark patterns in cookie consent for online news outlets.". In *In Proceedings of the 11th nordic conference on human-computer interaction: Shaping experiences, shaping societyy*, pages 1–12, 2020.

[22] IAB Eurppe. What is the transparency & consent framework (tcf)? https://iabeurope.eu/transparency-consent-framework/, 2019. Last accessed 26 March 2022.

[23] Irish Council for Civil Liberties. Gdpr enforcer rules that iab europe's consent popups are unlawful. https://www.iccl.ie/news/gdpr-enforcer-rules-that-iab-europes-consent-popups-are-unlawful/, 2022. Last accessed 27 March 2022.

[24] Google. Chrome devtools protocol - network domain - #method-getallcookies. https://chromedevtools.github.io/devtools-protocol/tot/Network/#method-getAllCookies, 2022. Last accessed 27 March 2022.

[25] Google. Clear browsing data. https://support.google.com/chrome/answer/2392709?hl=en&co=GENIE.Platform=Desktop, 2022. Last accessed 27 March 2022.

[26] Google. Google translate. https://translate.google.com/, 2022. Last accessed 27 March 2022.

[27] C.M. et al Gray. The dark (patterns) side of ux design. In *In Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems (pp. 1-14).*, pages 1–14, 2018.

[28] SuHun Han. pypi googletrans 3.0.0. https://pypi.org/project/googletrans/, 2020. Last accessed 27 March 2022.

[29] Kaspersky. What are cookies? https://www.kaspersky.com/resource-center/definitions/cookies, 2021. Last accessed 15 October 2021.

[30] et al. Kincaid, J. Peter. Derivation of new readability formulas (automated readability index, fog count and flesch reading ease formula) for navy enlisted personnel. In *Naval Technical Training Command Millington TN Research Branch*, 1975.

[31] Daniel Kladnik. I don't care about cookies 3.3.8 get rid of cookie warnings from almost all websites! https://www.i-dont-care-about-cookies.eu/, 2021. Last accessed 27 March 2022.

[32] Richie Koch. Cookies, the gdpr, and the eprivacy directive. https://gdpr.eu/cookies/, 2019. Last accessed 15 October 2021.

[33] FortiGuard Labs. Web filter lookup. https://www.fortiguard.com/webfilter, 2022. Last accessed 27 March 2022.

[34] Microsoft. P3p is no longer supported. https://docs.microsoft.com/en-us/previous-versions/windows/internet-explorer/ie-developer/compatibility/mt146424(v=vs.85), 2016. Last accessed 30 March 2022.

[35] Microsoft. Get your document's readability and level statistics. https://support.microsoft.com/en-us/office/get-your-document-s-readability-and-level-statistics-85b4969e-e80a-4777-8dd3-f7fc3c8b3fd2, 2021. Last accessed 8 Mat 2022.

[36] B. Molnar. Measuring the cookie-setting behaviour of web pages showing privacy warnings). 2020.

[37] Mozilla. ¡iframe¿: The inline frame element. https://developer.mozilla.org/en-US/docs/Web/HTML/Element/iframe, 2021. Last accessed 27 March 2022.

[38] Mozilla. Same-origin policy. https://developer.mozilla.org/en-US/docs/Web/Security/Same-origin_policy, 2021. Last accessed 15 October 2021.

[39] Mozilla. Using http cookies. https://developer.mozilla.org/en-US/docs/Web/HTTP/Cookies, 2021. Last accessed 15 October 2021.

[40] Mozilla. Comparing firefox browser with google chrome. https://www.mozilla.org/en-US/firefox/browsers/compare/chrome/, 2022. Last accessed 27 March 2022.

[41] Mozilla. Using shadow dom. https://developer.mozilla.org/en-US/docs/Web/Web_Components/Using_shadow_DOM, 2022. Last accessed 26 March 2022.

[42] Information Commissioners Office. Cookies and similar technologies. https://ico.org.uk/for-organisations/guide-to-pecr/cookies-and-similar-technologies/, 2021. Last accessed 15 October 2021.

[43] Information Commissioners Office. How do the cookie rules relate to the gdpr? https://ico.org.uk/for-organisations/guide-to-pecr/guidance-on-the-use-of-cookies-and-similar-technologies/how-do-the-cookie-rules-relate-to-the-gdpr/, 2021. Last accessed 15 October 2021.

[44] Information Commissioners Office. What are pecr? https://ico.org.uk/for-organisations/guide-to-pecr/what-are-pecr/, 2021. Last accessed 15 October 2021.

[45] OpenCV. pypi opencv. https://pypi.org/project/opencv-python/, 2022. Last accessed 27 March 2022.

[46] Rick Petnel. Easylist. https://easylist.to/, 2022. Last accessed 27 March 2022.

[47] Smilte Petronyte. Measure the cookie setting behavior of web pages showing cookie privacy warnings. 2021.

[48] Leonard Richardson. pypi beautifulsoup4 4.10.0. https://pypi.org/project/beautifulsoup4, 2021. Last accessed 27 March 2022.

[49] Ari Schwartz. Looking back at p3p: Lessons for the future. https://cdt.org/wp-content/uploads/pdfs/P3P_Retro_Final_0.pdf, 2009. Last accessed 30 March 2022.

[50] Simon Sharwood. Netscape navigator - the browser that started it all - turns 20. https://www.theregister.com/2014/10/14/netscape_navigator_the_browser_that_started_it_all_turns_20/, 2014. Last accessed 15 October 2021.

[51] Steven Vaughan-Nichols. What's the most popular web browser in 2021? https://www.zdnet.com/article/most-popular-web-browser-in-2021/, 2021. Last accessed 27 March 2022.

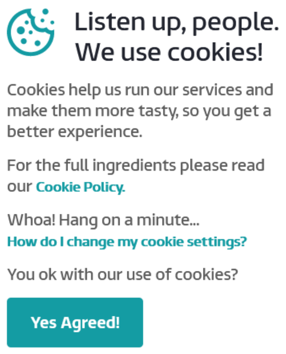
[52] Luis von Ahn et al. Captcha: Using hard ai problems for security. In *International conference on the theory and applications of cryptographic techniques*, pages 294–311, Springer, Berlin, Heidelberg., 2003.

[53] W3. Platform for privacy preferences (p3p) project. https://www.w3.org/P3P/, 2002. Last accessed 30 March 2022.

Appendix A

First appendix

A.1 Dark Pattern Examples



Listen up, people.
We use cookies!

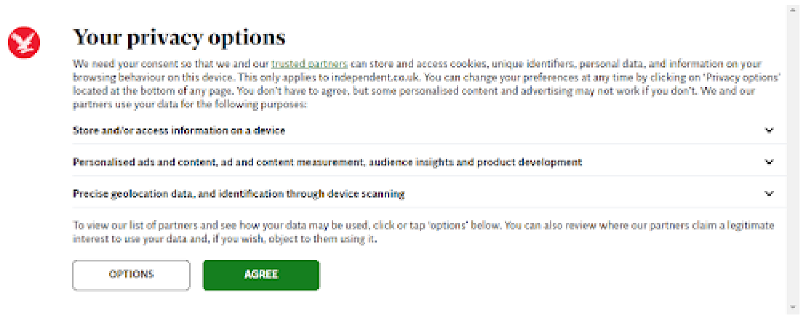
Cookies help us run our services and make them more tasty, so you get a better experience.

For the full ingredients please read our Cookie Policy.

Whoa! Hang on a minute...
How do I change my cookie settings?

You ok with our use of cookies?

Figure A.1: Example of Dark Pattern 1: Only Opt-in option is present on initial Cookie Dialog (Source: itv.com)



Your privacy options

We need your consent so that we and our trusted partners can store and access cookies, unique identifiers, personal data, and information on your browsing behaviour on this device. This only applies to independent.co.uk. You can change your preferences at any time by clicking on 'Privacy options' located at the bottom of any page. You don't have to agree, but some personalised content and advertising may not work if you don't. We and our partners use your data for the following purposes:

Store and/or access information on a device

Personalised ads and content, ad and content measurement, audience insights and product development

Precise geolocation data, and identification through device scanning

To view our list of partners and see how your data may be used, click or tap 'options' below. You can also review where our partners claim a legitimate interest to use your data and, if you wish, object to them using it.

Figure A.2: Example of Dark Pattern 2: Background colour of Opt-in button leads to it being highlighted more compared to Opt-out button (Source: independent.co.uk)

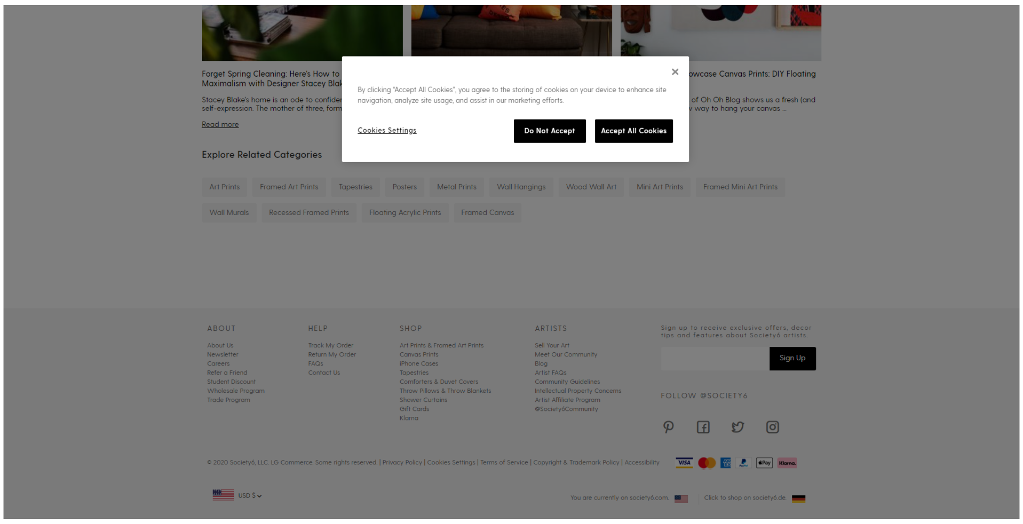
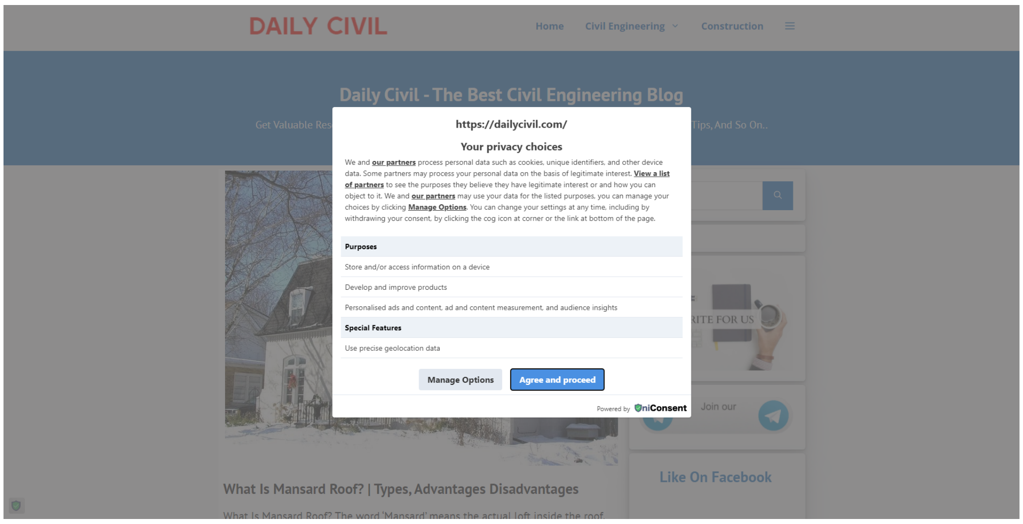

By clicking "Accept All Cookies", you agree to the storing of cookies on your device to enhance site navigation, analyze site usage, and assist in our marketing efforts.

Figure A.3: Example of Dark Pattern 3: Dialog Obstructs the window (Source: webstaqram.com)


https://dailycivil.com/

Your privacy choices

We and **our partners** process personal data such as cookies, unique identifiers, and other device data. Some partners may process your personal data on the basis of legitimate interest. **View a list of partners** to see the purposes they believe they have legitimate interest or and how you can object to it. We and **our partners** may use your data for the listed purposes, you can manage your choices by clicking **Manage Options**. You can change your settings at any time, including by withdrawing your consent, by clicking the cog icon at corner or the link at bottom of the page.

Purposes

Store and/or access information on a device

Develop and improve products

Personalised ads and content, ad and content measurement, and audience insights

Special Features

Use precise geolocation data

Figure A.4: Example of Dark Pattern 4: Text Content of the dialog is difficult to read, this has an FK score of 47.62 (Source: dailycivil.com)

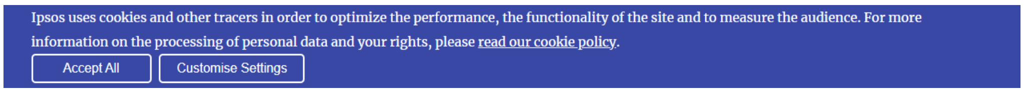
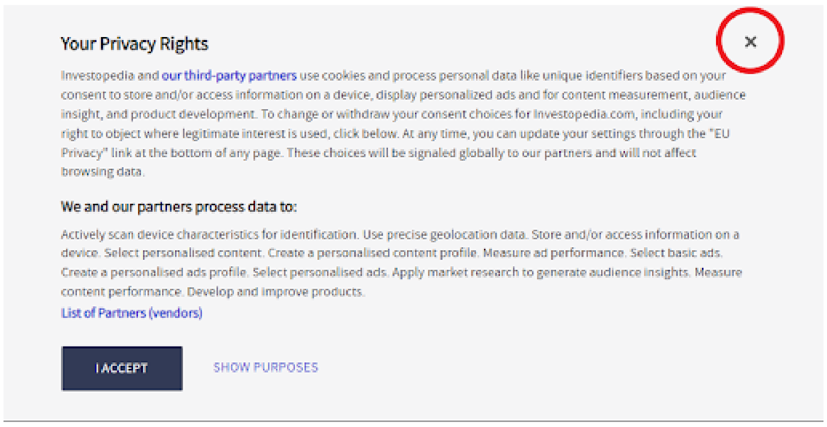

Ipsos uses cookies and other tracers in order to optimize the performance, the functionality of the site and to measure the audience. For more information on the processing of personal data and your rights, please read our cookie policy.

Figure A.5: Example of Dark Pattern 5: Multiple layers to a cookie dialog (Source: ipsos.com)



Your Privacy Rights

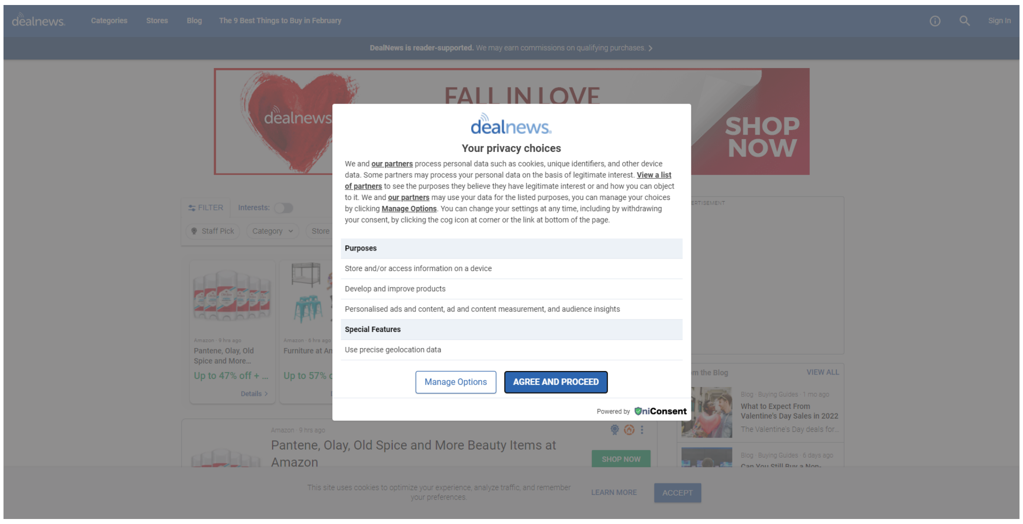
Investopedia and our third-party partners use cookies and process personal data like unique identifiers based on your consent to store and/or access information on a device, display personalized ads and for content measurement, audience insight, and product development. To change or withdraw your consent choices for Investopedia.com, including your right to object where legitimate interest is used, click below. At any time, you can update your settings through the "EU Privacy" link at the bottom of any page. These choices will be signaled globally to our partners and will not affect browsing data.

We and our partners process data to:

Actively scan device characteristics for identification. Use precise geolocation data. Store and/or access information on a device. Select personalised content. Create a personalised content profile. Measure ad performance. Select basic ads. Create a personalised ads profile. Select personalised ads. Apply market research to generate audience insights. Measure content performance. Develop and improve products.

List of Partners (vendors)

Figure A.6: Example of Dark Pattern 6: Ambiguous Close button is present on the dialog (Source: investopedia.com)



DealNews is reader-supported. We may earn commissions on qualifying purchases.

Your privacy choices

We and our partners process personal data such as cookies, unique identifiers, and other device data. Some partners may process your personal data on the basis of legitimate interest. View a list of partners to see the purposes they believe they have legitimate interest or and how you can object to it. We and our partners may use your data for the listed purposes, you can manage your choices by clicking Manage Options. You can change your settings at any time, including by withdrawing your consent, by clicking the cog icon at corner or the link at bottom of the page.

Purposes

Store and/or access information on a device

Develop and improve products

Personalised ads and content, ad and content measurement, and audience insights

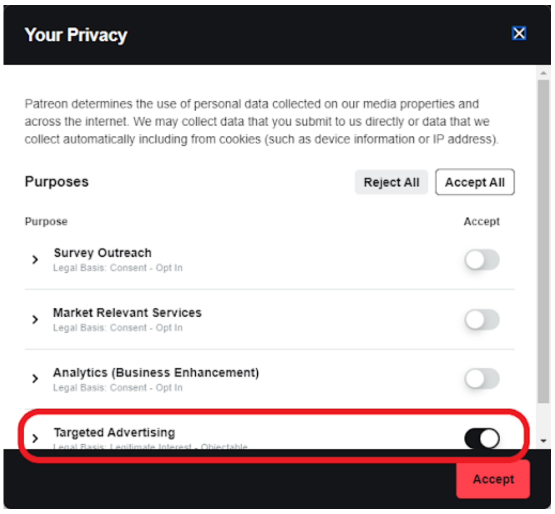
Special Features

Use precise geolocation data

Powered by UniConsent

This site uses cookies to optimize your experience, analyze traffic, and remember your preferences.

Figure A.7: Example of Dark Pattern 7: Multiple distinct Cookie Dialogs present on a page (Source: dealnews.com)

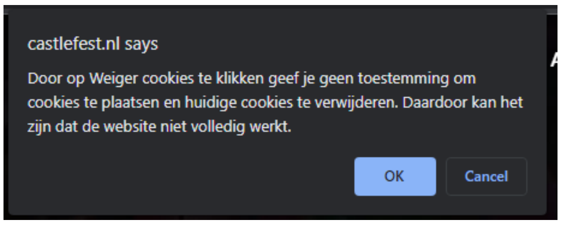

Your Privacy

Patreon determines the use of personal data collected on our media properties and across the internet. We may collect data that you submit to us directly or data that we collect automatically including from cookies (such as device information or IP address).

Purposes

Purpose	Accept
Survey Outreach Legal Basis: Consent - Opt In	Off
Market Relevant Services Legal Basis: Consent - Opt In	Off
Analytics (Business Enhancement) Legal Basis: Consent - Opt In	Off
Targeted Advertising Legal Basis: Legitimate Interest - Objectable	On

Figure A.8: Example of Dark Pattern 8: At least one Preference Slider is enabled by default (Source patreon.com).


castlefest.nl says

Door op Weiger cookies te klikken geef je geen toestemming om cookies te plaatsen en huidige cookies te verwijderen. Daardoor kan het zijn dat de website niet volledig werkt.

Figure A.9: Example of Dark Pattern 15: When a user clicks Opt-out button they are asked to confirm their choice (Source castlefest.nl).

A.2 Additional Data Set Details

The full list of domains used in the Random 500 and Random 10k data sets will be included as part of my submission of this report.

A.3 Database Schema

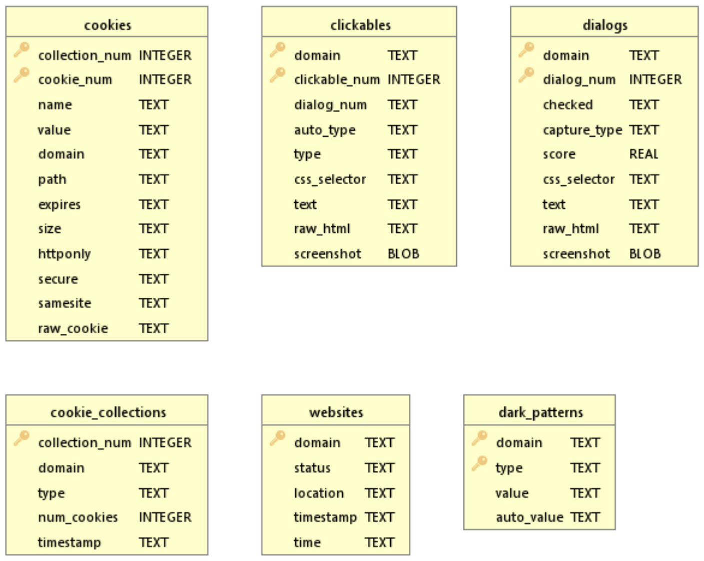


Figure A.10: Schema of the SQLite database used to store results from the system.

A.4 Dialog Ngrams

Uni-grams:

```
cookies
cookie
track
tracking
```

Bi-grams:

```
use cookies
cookies and
cookies to
we use
accept all
any time
at any
you agree
learn more
manage preferences
```

Tri-grams:

```
we use cookies
at any time
use cookies and
use cookies to
```

```
cookies and similar
use of cookies
learn more about
and our partners
and similar technologies
our cookie policy
```

4-grams:

```
we use cookies to
use cookies and similar
cookies and similar technologies
you can change your
access information on a
and or access information
at any time by
information on a device
or access information on
store and or access
```

5-grams:

```
access information on a device
and or access information on
store and or access information
use cookies and similar technologies
ad and content measurement audience
and content measurement audience insights
audience insights and product development
content measurement audience insights and
improve your experience on our
measurement audience insights and product
```

A.5 Custom Dialog CSS Selectors

```
div[class*="gdpr"]
div[class*="Cookie"]
div[class*="cookie"]
div[class*="Privacy"]
div[class*="privacy"]
div[class*="Policy"]
div[class*="policy"]
div[class*="Consent"]
div[class*="consent"]
div[class*="Notice"]
div[class*="notice"]
div[class*="Dialog"]
div[class*="dialog"]
```

```
div[id*="gdpr"]
div[id*="Cookie"]
div[id*="cookie"]
div[id*="Privacy"]
div[id*="privacy"]
div[id*="Policy"]
div[id*="policy"]
div[id*="Consent"]
div[id*="consent"]
div[id*="Notice"]
div[id*="notice"]
div[id*="Dialog"]
div[id*="dialog"]
div[data-project*="cmp"]
div[id*="privacy"]
div[id*="Privacy"]
div[id*="cmp"]
```

A.6 Custom Clickable CSS Selectors

A.6.1 General Clickable Selectors

```
button
a
[role='button']
input[type='submit']
input[type="checkbox"]
span[class*="button"]
span[class*="Button"]
span[class*="btn"]
span[class*="btn"]
[class*="close"]
[class*="Close"]
[class*="button"]
[class*="Button"]
[data-tracking-opt-in-learn-more]
[data-tracking-opt-in-accept]
[class*="settings"]
[id*="custom"]
[id*="accept"]
[class*="settings"]
[class*="custom"]
[class*="accept"]
```

A.6.2 Close Button Clickable Selectors

```
[class*="close"]
[class*="Close"]
[aria-label*="close"]
[aria-label*="Close"]
svg
```

A.6.3 Preference Slider Clickable Selectors

```
input[type="checkbox"]
```

A.7 Clickable Keyword Combinations

Opt-in		**Opt-out**		**more options**
¬ (no ∨ no ∨ don't) ∧	accept	Opt-in keyword ∧	essenti	adjust
	activ		necessari	advanc
	agre		requir	choic
	allow	essenti	∨ cooki onli	configur
	assent	necessari		custom
	confirm	requir		customis
	consent	(not ∨ no ∨ don't) ∧	accept	manag
	continu		consent	option
	enabl		track	personali
	enter	disagre		prefer
	fine	reject		purpos
	ok	declin		set
	okay	refus		tool
	opt-in	deactiv		let me choos
	proceed	continu without accept		select cooki
	understand			
	understood			
	yes			
got it				
i am happi with all cooki				

Table A.1: keyword Combinations for Opt-in, Opt-out and more options clickables.

Preference Slider	**Confirm Preferences**		**Close Option**	**Policy Link**
on	save		close	privaci
off	submit		dismiss	polici
	Opt-in keyword ∧	select	exit	notic
		choic	x	here
		custom	×	cooki
		prefer	hide	vendor
				partner
				use of cooki
				data protect
				terms servic
				read more
				learn more
				tell me more
				more inform
				see detail
				more detail

Table A.2: keyword Combinations for Preference Slider, Confirm Preferences, Close Option and Policy Link clickables.