Are they Toeing the Line? Diagnosing Privacy Compliance Violations among Browser Extensions

Yuxi Ling
National University of Singapore

Kailong Wang*
Huazhong University of Science and Technology
National University of Singapore

Guangdong Bai†
The University of Queensland

Haoyu Wang†
Huazhong University of Science and Technology

Jin Song Dong
National University of Singapore

ABSTRACT

Browser extensions have emerged as integrated characteristics in modern browsers, with the aim to boost the online browsing experience. Their advantageous position between a user and the Internet endows them with easy access to the user's sensitive data, which has raised mounting privacy concerns from both legislators and extension users. In this work, we propose an end-to-end approach to automatically diagnosing the privacy compliance violations among extensions. It analyzes the compliance of privacy policy versus regulation requirements and their actual privacy-related practices during runtime. This approach can serve the extension users, developers and store operators as an efficient and practical detection mechanism for privacy compliance violations.

Our approach utilizes the state-of-the-art language processing model BERT for annotating the policy texts, and a hybrid technique to analyze an extension's source code and runtime behavior. To facilitate the model training, we construct a corpus named PrivAud-100 which contains 100 manually annotated privacy policies. Our large-scale diagnostic evaluation reveals that the vast majority of existing extensions suffer from privacy non-compliance issues. Around 92% of them have at least one violation of either their privacy policies or data collection practices. Based on our findings, we further propose an index to facilitate the filtering and identification of privacy-incompliant extensions with high accuracy (over 90%). Our work should raise the awareness of extension users, service providers, and platform operators, and encourage them to implement solutions toward better privacy compliance. To facilitate future research in this area, we have released our dataset, corpus and analyzer.

ACM Reference Format:
Yuxi Ling, Kailong Wang, Guangdong Bai, Haoyu Wang, and Jin Song Dong. 2022. Are they Toeing the Line? Diagnosing Privacy Compliance Violations among Browser Extensions. In *37th IEEE/ACM International Conference on Automated Software Engineering (ASE '22), October 10–14, 2022, Rochester, MI, USA.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3551349.3560436

1 INTRODUCTION

Browser extensions (extensions for short hereafter), which are installed as third-party software modules to enrich the functionalities of browsers and optimize the user experience, have become a defining feature for modern browsers. Similar to other user-oriented applications such as web applications and mobile applications, the extensions are pivotal gateways connecting end-users and the diversified services online, which grant them easy access to sensitive user data. In particular, the basic and explicit personal identifiable information (i.e., name, email, address, etc.) and the users' online footprints (e.g., the contents they read, watch, click on and share) could be recorded and utilized to generate fine-grained user profiles.

Accompanied by the data collection and their subsequent handling (i.e., trading, processing, storage, and transfer), privacy concerns are raised by both legislators and Internet users. As a response, data protection regulations have been actively proposed and implemented around the world, with well-known examples being the General Data Protection Regulation (GDPR) [40] in European Union (EU), California Consumer Privacy Act (CCPA) [41] in California, Personal Data Protection Act 2012 (PDPA) [33] in Singapore and General Personal Data Protection Law (LGPD) [5] in Brazil. They provide general guidelines including what, when and how user data can be collected and processed.

Following the regulations, major platforms have enacted towards user data privacy. For example, Chrome and Firefox require the developers to provide a clear and concise summary of privacy practices (e.g., permissions requested and types of data collected) on each extension's introduction page, in addition to the full disclosure of the privacy policy. Unfortunately, such measures are far from sufficient to guarantee the actual compliance, as there is no reliable vetting process to ensure consistency from the guidelines to the extension's declared privacy policy and its actual practices.

Verifying such end-to-end privacy compliance among extensions is a sophisticated task. Referencing the relevant efforts in the literature [17, 20, 27, 45, 48, 50, 51], we have identified at least two following key challenges in this compliance diagnostic evaluation.

One key challenge is the precise mapping from an extension's privacy policy to the regulation requirements and its actual practices. It spans various aspects including data collection details (e.g., fine-grained data types, data collector's contact details, the purpose of collection, etc.) and user's data rights (e.g., right to access, rectify or erase the collected data). This mapping serves as the basis for verifying the completeness of user notification information in the privacy policy and the actual practice consistency at runtime. The intuitive approach for constructing the mapping is via training a

[*] Kailong Wang is the co-first author
[†] Guangdong Bai and Haoyu Wang are the corresponding authors

ASE '22, October 10–14, 2022, Rochester, MI, USA
© 2022 Copyright held by the owner/author(s).
ACM ISBN 978-1-4503-9475-8/22/10.
https://doi.org/10.1145/3551349.3560436

natural language processing (NLP) model. Nonetheless, there is an absence of a well-annotated corpus to comprehensively cover all the aspects during the model training.

The other key challenge is the efficient and universal approach to analyzing extensions' actual privacy-related practices, given their diversified characteristics (e.g., source file structures, user interfaces (UIs), DOM structures). The general idea is to extract the program behavioral information from the source code and runtime data. However, the commonly used program analysis techniques, such as those based on data flow and control flow analysis, have been deemed infeasible for a large-scale diagnostic evaluation [51]. In addition, the heterogeneity among extensions also hinders the implementation of a generic dynamic testing tool for all extensions.

Our work. In this paper, we propose a scalable end-to-end approach to automatically diagnosing the potential compliance violations versus the privacy regulations among browser extensions. We aim to provide an efficient and practical solution to the privacy compliance evaluation tasks for extension users, developers and store operators. In particular, for users, the proposed approach facilitates their understanding of privacy policies and provides the compliance report of an extension before downloading and installation. For developers, the proposed approach can scan for possible compliance violations in their new extensions before the release. For store operators, it facilitates the detection of existing compliance violations from on-shelf extensions, reducing the legal and operational costs.

Our approach consists of two major modules: *a policy compliance checker* and *a practice compliance checker*. The former utilizes a BERT model [9] trained on PrivAud-100, a corpus constructed in this work that contains 100 annotated privacy policies, to automatically annotate each sentence in a privacy policy. We consider a richer range of information during this process, including the regulation-mandated types of information and the finer-grained data types being collected. By examining the level of label completeness versus the privacy requirements, we quantitatively measure the privacy policy compliance. The latter module is a hybrid analysis tool that scans the privacy-related components from the source code (e.g., the API and permission requests) and during the runtime (e.g., the input fields dynamically generated). Furthermore, we propose an index to quantitatively measure the extension's likelihood to be privacy compliant/incompliant. To understand the privacy compliance status quo among extensions, we employ our approach for comprehensive screening against 64,114 extensions available in the Chrome Extension Store as of April 2022.

Contributions. To the best of our knowledge, this is the first work on diagnosing privacy compliance violations for browser extensions at scale. Our study reveals the prevalent poor privacy compliance from both privacy policies given by the developers and the actual extension practices in the runtime. In summary, this work mainly contributes to the following aspects.

- **A fully-automated approach.** We develop a fully-automated diagnostic approach for privacy compliance violations in browser extensions. It first annotates the privacy policy of extensions and reports the missing notification information required by GDPR. Then, it proceeds to analyze the privacy practices of an extension and flag those violating the declarations in the privacy policy.
- **A systematic large-scale study.** We investigate the status quo of extension privacy compliance by employing the proposed diagnostic approach to the extensions in our newly collected comprehensive dataset. We further propose an index to facilitate detecting privacy-incompliant extensions, which achieves a high accuracy of over 90%.
- **Practical Results.** We have collected and assembled the most comprehensive Chrome extension dataset, containing a total of 64,114 extensions. To facilitate annotating a wider range of information from privacy policy texts compared to the existing works [24, 51], we construct the corpus PrivAud-100 which contains 100 manually labeled privacy policies. Through our analysis, we reveal a shocking fact that only around 8% of the extensions are fully privacy compliant. We also identify prevalent discrepancies among privacy regulations, privacy policies, and practices for the vast majority of the extensions.

We release the Chrome extension dataset [7] and open source our analyzer [7] to facilitate future research in this area.

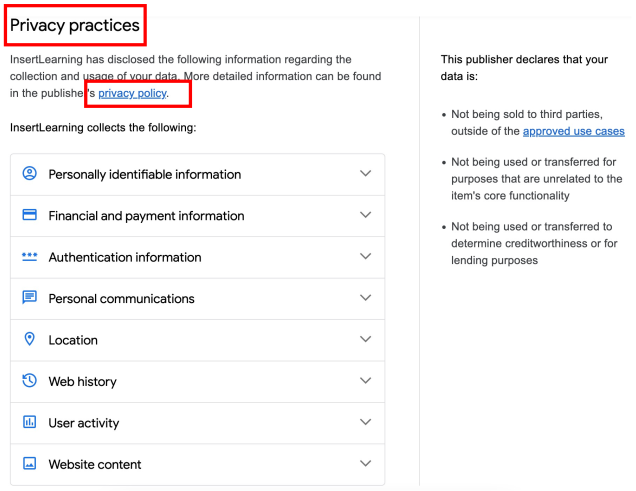
2 PRELIMINARIES

2.1 GDPR

GDPR has been enforced since May 2018 in the EU and the European Economic Area (EEA), superseding the Data Protection Directive 95/46/EC [1]. GDPR contains 11 chapters and 99 articles that regulate the data collection, usage, sharing, security, and processing practices of the data controllers (e.g., organizations), protecting data subjects (e.g., individual consumers) residing in the EU and EEA regardless of the location of the data controllers. In particular, Article 13 details the information that has to be provided to the data subjects once their personal data is collected. As such information is commonly reflected in the privacy policies, we focus on Article 13 and use the requirements extracted from it [24] to empower further analyses and detection of the GDPR violations from the extensions developed by data controllers.

2.2 Privacy Policy and Privacy Practice Summary (PPS)

A service operator (e.g., a website or software) is required to provide users with a privacy policy as a statement or legal document, notifying the consumer data practices such as data collection, handling, and processing. To further complement the privacy policy, major browser extension platforms further request developers to provide extra information regarding their actual runtime behaviors. In particular, the privacy practice summary (PPS) page mandated by Chrome Extension Store lists the detailed data collection practices. For example, the extension *InsertLearning* [11] declares to collect various personally identifiable information, such as device information and user activity, as shown in Figure 1. At the same time, this extension also declares the limited use of the collected data by stating that the collected user data will not be sold to third parties, be used outside the core function of this extension, or be used for lending purposes.



Privacy practices

InsertLearning has disclosed the following information regarding the collection and usage of your data. More detailed information can be found in the publisher's privacy policy.

InsertLearning collects the following:

- Personally identifiable information
- Financial and payment information
- Authentication information
- Personal communications
- Location
- Web history
- User activity
- Website content

This publisher declares that your data is:

- Not being sold to third parties, outside of the approved use cases
- Not being used or transferred for purposes that are unrelated to the item's core functionality
- Not being used or transferred to determine creditworthiness or for lending purposes

Figure 1: Privacy Practices of InsertLearning on Chrome Extension Store

2.3 Behavior-related Extension Features

Before we can analyze the actual privacy practices, we briefly introduce the two key categories of behavior-related features in extensions.

Requested Permissions and API Usage. Extensions are commonly archived into a *crx* or *xpi* file, inside which the source files such as HTML, JavaScript, JSON, and local images are organized in a similar way to a web application. In particular, the requested permissions listed in the *manifest.json* file determine an extension's capability to access and handle the user data. For example, the `Storage` permission enables storing, retrieving, and tracking of changes to a user's data. Furthermore, API calls granted under certain permission reveal finer-grained information on the actions or functions performed. For example, the `chrome.storage.local` stores the user data into the LocalStorage. Thus, they serve as a critical indicator of the extension's privacy practices.

HTML and DOM Features. The execution and functionality of each extension are detailed in the JavaScript source code. Due to the dynamically rendered contents and heterogeneity among the source code, analyzing and identifying a particular element (e.g., a privacy-relevant element) of an extension using traditional static analysis (e.g., control flow and data flow analysis) becomes complicated and ineffective. As an alternative, the Document Object Model (DOM) tree of a dynamically generated HTML file provides an accurate hierarchical representation of the available objects on the extension interface, allowing us to directly query those related to the extension's privacy practices.

3 DIAGNOSING VIOLATIONS IN PRIVACY POLICIES AND PRACTICES

3.1 Problem Formalization

Privacy Requirements. Based on the clauses stated in the GDPR Article 13, the data controllers must satisfy the minimum privacy requirements $\Upsilon = \{R_1, R_2, \cdots, R_j\}$ once they collect the user data, where R_i $(1 \leq i \leq j)$ denotes a label representing a type of privacy notification information to the data subjects. We use a minimum set of required information in this paper, as listed in Table 1.

Table 1: Privacy Notification Information Categories Extracted from GDPR Article 13 [23]

Privacy Notification	Explanation
Data collection (**DC**)	Provide accurate categories of personal identifiable information collected from data subjects
Data retention (**DR**)	Retention period of the collected data
Data usage (**DU**)	Purpose of data processing and usage in the products and services
Contact information (**CI**)	Provide contact details of the controller or the data protection officer
Data subjects rights(**DSR**)	Rights to access, rectify or erase, restrict of processing, object to processing, data portability and lodge complaints

Table 2: Data Types Collected in Privacy Practices

Data Types	Explanation
Personal information (**PI**)	Personal identifiable and authentication information, and user-specific files saved to the browser storage
Activity information (**AI**)	User browsing-related data
Device information (**DI**)	Hardware runtime status data of the user device

When the label **DC** is in a privacy policy, the data controller must provide the specific personal data types $\Delta = \{D_1, D_2, \cdots, D_k\}$ being collected during the extension usage, where D_i $(1 \leq i \leq k)$ denotes a type of personal data. The full list of data types is summarized in Table 2.

Privacy Policies and Practices. Given an extension, the data subject notification information in its privacy policy is abstracted as $\Pi = \{N_1, N_2, \cdots, N_p\}$, and the actual data collection practices are abstracted as $\Gamma = \{P_1, P_2, \cdots, P_q\}$. In particular, $\Pi \subseteq \Upsilon$, and $\Gamma \subseteq \Delta$.

Problem Statement. The privacy compliance diagnostic evaluation can be defined as the verification:

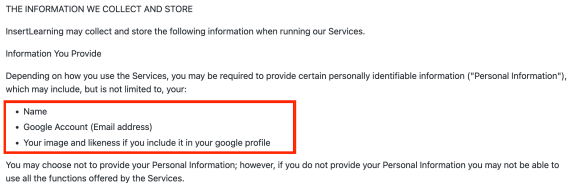
- For *policies*: check if $\Pi = \Upsilon$
- For *practices*: when the label "**DC**"$\in \Pi$, check if $\Gamma = \Delta$

Intuitively, we aim to analyze the comprehensiveness of elements in set Π versus those in set Υ (i.e., *privacy policy compliance*), and the comprehensiveness of elements in set Γ extracted from each extension versus those in set Π (i.e., *privacy practice compliance*). Note that we focus on the data collection practices as other aspects in data management (e.g., data retention and data subject rights) are conducted on the service provider end, which is not directly relevant to the extension's behaviors. We will include more discussion on this issue in Section 6.

3.2 A Sample Diagnostic Evaluation

Based on our problem definition, we use the extension *InserLearning* as an example to illustrate the highlighted aspects of our evaluation. We further provide two detailed examples in our online repository [7]: one is privacy compliant and the other is incompliant (or partially compliant).

Privacy Policy Compliance. We first read the complete privacy policy provided by the developer (available at the *privacy policy* link in Figure 1). Then, we manually annotate each sentence based on its semantics related to the privacy notification category in Table 1, and consult experts from law school to ensure the annotation correctness. By aggregating the labels for all the sentences, we derive the summarized privacy policy Π_{IL}:



THE INFORMATION WE COLLECT AND STORE

InsertLearning may collect and store the following information when running our Services.

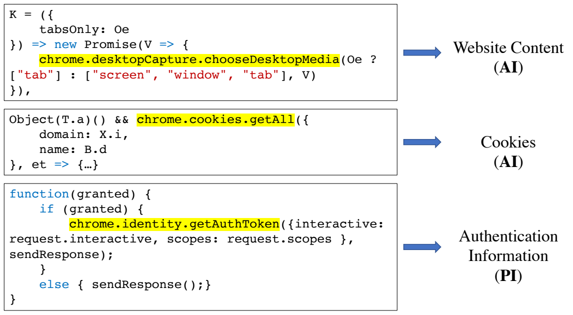
Information You Provide

Depending on how you use the Services, you may be required to provide certain personally identifiable information ("Personal Information"), which may include, but is not limited to, your:

- Name
- Google Account (Email address)
- Your image and likeness if you include it in your google profile

You may choose not to provide your Personal Information; however, if you do not provide your Personal Information you may not be able to use all the functions offered by the Services.

Figure 2: Personal Identifiable Information Collected by Extension *InserLearning*



```
K = ({
    tabsOnly: Oe
}) => new Promise(V => {
    chrome.desktopCapture.chooseDesktopMedia(Oe ?
["tab"] : ["screen", "window", "tab"], V)
}),
```

```
Object(T.a)() && chrome.cookies.getAll({
    domain: X.i,
    name: B.d
}, et => {…}
```

```
function(granted) {
    if (granted) {
        chrome.identity.getAuthToken({interactive:
request.interactive, scopes: request.scopes },
sendResponse);
    }
    else { sendResponse();}
}
```

Figure 3: An example of the projection from Chrome APIs to private user data collection, from extension *InserLearning*

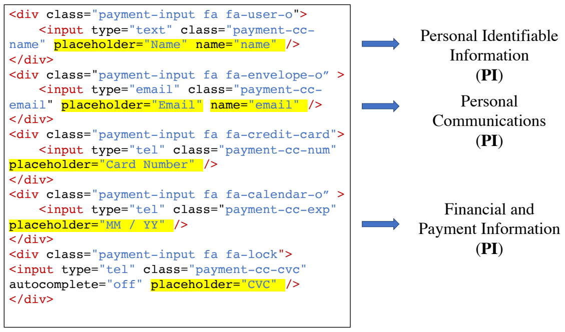
$$\Pi_{IL}=\{\textbf{DC}, \textbf{DU}, \textbf{CI}\}$$

We find that *InserLearning* fails to provide information on data retention and user rights. In addition, we also notice the discrepancies between the declared information from its PPS page (i.e., 4 types of personal data as listed in Figure 1) and that in its privacy policy (i.e., only 1 type of personal data as highlighted in Figure 2).

Privacy Practice Compliance. We examine the source code of the extension, identify the data collection behaviors, and compare them with those declared in the privacy policy. In particular, we focus on the data collection practices.

We manually inspect the following two types of information. First, we check the Chrome API calls as they directly implement the functionality of an extension, which consequently determines its data collection practices. As shown in Figure 3, we discover them by string matching function names starting with "`chrome`". From them, we can infer the collected personal data. For example, the `chrome.cookies.getAll` suggest the access and collection of the user cookies in the browser. Second, we record the HTML files dynamically generated after clicking each button on the UI, and search for user input-related HTML tags (e.g., `<input>`) to identify the types of collected user data based on their attributes (e.g., `placeholder`), as shown in Figure 4. In this way, we can efficiently extract the user input data without the complication of constructing the control flow and data flow information.

We find that *InserLearning* collects more user data in practice compared to that in the privacy policy. For example, the information on financial payments (e.g., credit card number, expiry date, and CVC code) and user authentication. Meanwhile, we also notice an "over-claim" of the collected data compared to declarations on the PPS page where the location information is declared but not collected.



```
<div class="payment-input fa fa-user-o">
    <input type="text" class="payment-cc-
name" placeholder="Name" name="name" />
</div>
<div class="payment-input fa fa-envelope-o” >
    <input type="email" class="payment-cc-
email" placeholder="Email" name="email" />
</div>
<div class="payment-input fa fa-credit-card">
    <input type="tel" class="payment-cc-num"
placeholder="Card Number" />
</div>
<div class="payment-input fa fa-calendar-o” >
    <input type="tel" class="payment-cc-exp"
placeholder="MM / YY" />
</div>
<div class="payment-input fa fa-lock">
<input type="tel" class="payment-cc-cvc"
autocomplete="off" placeholder="CVC" />
</div>
```

Figure 4: An example of the projection from HTML tags to private user data collection, from extension *InserLearning*

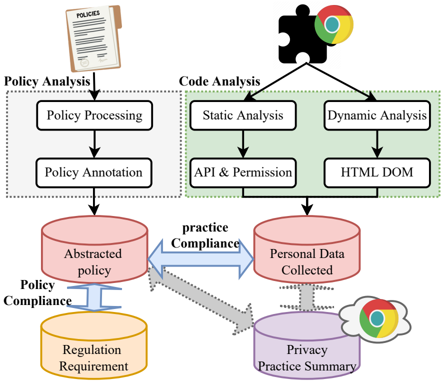


Figure 5: Methodology Overview

3.3 Methodology

3.3.1 Methodology Overview. To expedite the diagnostic evaluation following our manual inspection, we propose a fully automatic analysis approach, leveraging techniques of NLP and hybrid program analysis. The working flow of our approach is illustrated in Figure 5, which consists of two modules: **(1) A privacy policy compliance checker.** This module extracts a set of user notification information from the sentences of a privacy policy text, and compares this set with that mandated by the privacy regulations. **(2) A privacy practice compliance checker.** This module utilizes the hybrid code analysis to capture the privacy-centric data collection practices of an extension, focusing on API usage and user-oriented functionalities. As a part of our comprehensive evaluation, we also check the platform-mandated declarations on the PPS pages declared by the developers (marked as grey arrows in Figure 5).

Challenges. To implement our approach, we have identified the following three key challenges:

- **Challenge#1.** There is a lack of a privacy policy corpus that is comprehensively annotated to facilitate the end-to-end privacy compliance diagnostic evaluation. It should cover the notification information types required by GDPR and the detailed data types collected by the actual privacy practices. The existing corpora in the literature only contain partial information types we target.

- **Challenge#2.** There is a lack of an effective method to annotate the privacy policies for the diagnostic evaluation at scale. Given that existing works utilize NLP models trained on specific corpora, we find it necessary to leverage state-of-the-art algorithms to implement and optimize our own model for the annotation task.
- **Challenge#3.** There is a lack of a lightweight method to analyze the data collection practices from extensions at scale. The widely-used program analysis techniques in the literature (e.g., control flow analyses) are regarded as infeasible for a large-scale study due to their complexity.

3.3.2 Corpus PrivAud-100 (Solution to Challenge#1). We construct our corpus PrivAud-100 which consists of 100 randomly selected privacy policies from Chrome extensions. We first preprocess them into sentences. In more detail, we use BeautifulSoup4 [4] to extract text strings from the HTML files on the privacy policy web pages. Then, we remove the non-ASCII symbols and use regular expression pattern matching to further split the policy into single sentences. After preprocessing, we produce a `.tsv` file where each row contains one policy sentence, a privacy policy ID, and a label denoting the sentence category. The default label for each sentence is assigned as *Other Info*. During the manual annotation process, a new label will be re-assigned to a sentence once it contains target information.

The manual annotation is conducted by two of the co-authors who carefully check the content and context of each sentence based on their understanding. They both have research and education backgrounds related to privacy regulations. We use the Percent Agreement [16] to measure the inter-rater reliability between them, which shows a high agreement (0.96) on all the sentences annotated. For the 4% sentences with label disagreements, we consult an expert from law school to reach a consensus. During the consultation, we also randomly sample 3% of the sentences to confirm the correctness of the agreed labels. Among them, we find no mislabeled sentences.

Note that we adopt a more comprehensive set of labeling rules compared to the existing works [24, 51], as we aim to check both the policy and practice compliance. In particular, we check the policy compliance with labels listed in Table 1. Meanwhile, we label the fine-grained types of user data once they are collected by an extension (i.e., the label **DC** is set in the privacy policy), as shown in Table 2. We specifically consider personal information (e.g., name, address, date of birth, financial information), user activity information (e.g., content browsed, location), and device information (e.g., CPU runtime status). Our corpus is available in the repository [7].

3.3.3 Privacy Policy Compliance Checker (Solution to Challenge#2). The general idea is to train a classification model on our corpus PrivAud-100, and use its prediction results to annotate the text in a privacy policy. It is regarded as privacy-compliant if all types of notification information required by GDPR are present.

To this end, we train and select a suitable classifier from three machine learning algorithms commonly used in NLP tasks including SVM [8], Bidirectional LSTM (BiLSTM) [15], and BERT [9], using our corpus PrivAud-100 as ground truth. We utilize the best-performing trained classifier to predict the labels of the rest preprocessed sentences, and check the label completeness for each privacy policy versus Table 1. To further evaluate the robustness of our models, we utilize the available corpora in the literature [24, 51] that partially cover the labels in PrivAud-100. More specifically, we create a mapping table (i.e., Table 3) to transform labels of the corpora into those used in PrivAud-100. In this way, we can use the corpus [24] to test the prediction results for the notification types (e.g., "Data retention", "Contact details", etc.), and use the corpus APP-350 [51] to test the prediction results for the fine-grained data types to be collected ("Address", "Cookies", etc.). We will detail the evaluation in Section 5.

Table 3: Label Mappings

Corpus	New Labels	Original Labels
Liu2021	**DC**	Collect personal information
	DR	Data retention
	DU	Data processing purpose
	CI	Contact details
	DSR	Right to access, to rectify/erase, to restrict/object to processing, to data portability, to lodge a complaint
APP-350	**PI**	Contact: address book, city, email, phone number, postal address, ZIP
		Authentication: SSO identifier
		Demographic: age, gender
	AI	Cookie or similar tech
	DI	Identifier: device ID, IMEI, IMSI, IP address, MAC, mobile carrier, SIM serial, SSID BSSID,
		Location: Bluetooth, cell tower, GPS, IP address, WiFi

3.3.4 Hybrid Source Code Analysis (Solution to Challenge#3). Considering the complexity of the commonly-used control flow and data flow analysis techniques on source code, we resort to a lightweight hybrid analysis to efficiently identify data collection practices from extensions on a large scale. To be consistent with the data types targeted in the policy analysis, we focus on the same types of user-related data listed in Table 2.

Static JavaScript code analysis. To identify Chrome APIs related to data collection practices, we mainly check their types and usage in the static analysis. First, we convert all the JavaScript files into Abstract Syntax Trees (ASTs) using *esprima* [10], which provide a clearer view of the function semantics and its usage compared to the plain source code. Second, we extract the API calls using keywords and pattern matching while traversing through the ASTs based on Algorithm 1. Intuitively, we first find the nodes with the name `chrome` through depth-first search (DFS). Then, we locate the boundary of the API call by repetitively traversing back to the parent nodes until one with the type `CallExpression` is reached. We collect the nodes' property names along the way to determine the personal data types. Third, based on the function argument semantics derived from the ASTs and the chrome browser official documents, we correlate the API usage to specific user data access/collection, as listed in Table 4.

Dynamic HTML pages analysis. To efficiently and precisely identify data collection practices during extension runtime, we focus on the HTML code components related to user data inputs (e.g., login credentials) and user operations (e.g., click a button).

First, we implement a tool based on *Selenium* [37] to dynamically simulates the user interactions with an extension. Utilizing it, we can automate the HTML file collection process. In more detail, the tool will sequentially install the downloaded extensions from our dataset. After each installation, it launches the extensions, scans

Algorithm 1: AST Query

```
ExtractChromeAPI ($T$);
Input   : $T$ : the AST object list in DFS sequence of a JS file;
          $Node$ : the node in the AST object
Output  : $S$ : the set of used chrome API and API's input
          arguments
$lastNode \leftarrow root$
foreach $Node \in T - root$ do
    $parentChain.push(lastNode)$
    // maintain a parent nodes stack
    if $Node.name$ is "chrome" then
        $functionName \leftarrow Null$
        while $parentChain \neq \emptyset$ do
            $parentNode = parentChain.pop()$
            $backupNodes.push(parentNode)$
            if $parentNode.type$ is "CallExpression" then
                $S.push(functionName, parentNode.arguments)$
                break // find the API in this branch
            else if $parentNode.type$ is "MemberExpression"
              then
                $functionName$ +=
                  $parentNode.property.name$
                //joint the intermediate function
            else
                break // unqualified
            end
        end
        push $backupNodes$ back to $parentChain$
    end
    $lastNode = Node$
end
return $S$
```

Table 4: Privacy-related chrome browser APIs

Data Types	Chrome API

PI	`chrome.fileBrowserHandler, chrome.storage`
	`chrome.identity, chrome.privacy`

AI	`chrome.accessbilityfeatures, chrome.browsingdata,`
	`chrome.contentSettings, chrome.declarativeNetRequest,`
	`chrome.desktopCapture,chrome.devtools, chrome.history,`
	`chrome.permissions, chrome.scripting, chrome.cookies`

DI	`chrome.enterprise,chrome.instanceID, chrome.power,`
	`chrome.proxy,chrome.runtime,chrome.system`

for user-interactive items (e.g., buttons, checkboxes) on the UI, and sequentially interacts with them to simulate the usage scenario from a user. Specifically, the *Register* button click will reveal the user input data fields for most of the personal information to be collected by the extension, as shown in Figure 4. Meanwhile, it records all the dynamically generated HTML files accompanied by the interactions.

Algorithm 2: DOM Tree Query

```
ExtractDOMElement ($D$);
Input   : $DT$ : the DOM tree element list of a HTML file
Output  : $S$ : the set of input related tags with attributes;
          $PD$ : private data collected in the HTML file
$targetEleList \leftarrow$ privacy-related DOM element names
$mapper \leftarrow$ mapper from keywords to data types
foreach $element \in DT$ do
    if $element.name \in targetEleList$ then
        $S.push(tag.name, tag.attributes\ )$
        foreach $attr \in tag.attributes.content$ do
            if $attr \in mapper.keys$ then
                $PD.push(mapper[attr])$
            end
        end
    end
end
return $S, PD$
```

Table 5: Privacy-related HTML tags

Data Types	Tags
User Inputs	`<input>`, `<fieldset>`, `<form>`, `<textarea>`
User Operations	`<button>`, `<select>`, `<optgroup>`, `<option>`, `<datalist>`

Table 6: Keywords List for DOM Tree Analysis

Data Types	Subtype	Keywords
PI	basic info	name, age, phone, email, id, account, password, pass word, pwd, seed phrases, backup words
	health info	symptom
	payment info	card number, CVV, expiry date
	location info	city, country, postal code, door number, street, building, community
	user document	upload, submit, select file, access files
DI	device info	mac address, ip address, device id
AI	user activity	privacy, accessibility, history, setting, preference

Next, we parse the HTML Document Object Model (DOM) trees following Algorithm 2 and extract the elements related to user inputs and operations (e.g., user input fields after clicking the *Register* button). Intuitively, the algorithm traverses through the DOM tree by breadth-first search (BFS). If a node's tag name is in the target element list as shown in Table 5, we will record this node and its attributes. After parsing the HTML files, we correlate each element to a certain data type by string matching their names against our summarized keywords list in Table 6. For efficiency concerns, we only traverse through the shallow layers (i.e., the first two layers) of an extension's UIs. We have conducted a validity study on this in Section 5.2.2 and further discuss its threat to validity in Section 6.

4 DATA COLLECTION

Before applying the proposed privacy compliance diagnostic approach, we first construct a comprehensive dataset of the extensions together with their privacy policies, as there is none to be reused for this study.

A Comprehensive Extension List. We target the Chrome extensions which cover around 80% of the global web browser market share in 2022 [14]. Considering the absence of official APIs for efficiently discovering the full list of extensions under each category, we start with the Chrome extension names from a dataset collected in 2021 [22] and use each of them as a set of keywords to further expand the search in the Chrome Extension Store. We keep the top 20 results under each search and aggregate the final lists by removing the duplicated items. We manage to obtain a total of 64,114 names of extensions. To the best of our knowledge, this is the most comprehensive list, if not complete, for Chrome extensions collected from 2022.

Collection Methodology. To collect the source code and the privacy policies of the extensions from our list, we deploy our crawler on one Ubuntu 18.04 virtual machine equipped with two Xeon Silver 4108 CPUs, 128 GB RAM, and 960 GB storage. To speed up the collection, we utilize 20 threads in the crawler. We manage to complete a round of collection within 42 hours, which renders it feasible for real-time monitoring on a regular basis. For the source code, we have successfully crawled from all of the 64,114 extensions, with a total of 163 GB data. For the privacy policies, we have collected from 20,761 Chrome extensions, due to the prevalent absence of privacy policy URLs. This indicates the poor privacy policy compliance among browser extensions, which we will detail in Section 5.

5 EVALUATION

After collecting the source code and privacy policy files, we proceed to examine their privacy compliance against GDPR using the methodology detailed in Section 3. Our investigation targets the following three research questions (RQs).

RQ1. What is the effectiveness of our methodology? This research question aims to understand the accuracy and robustness of our approach.

RQ2. What is the state of privacy compliance of extensions? This research question aims to measure the degree of end-to-end privacy compliance in the ecosystem of Chrome extensions, focusing on both privacy policy and privacy practice compliance.

RQ3. What are the features of the privacy (in)compliant extensions? This research question aims to identify the characteristics that can be utilized to further flag extensions that have a high risk of privacy incompliance.

5.1 RQ1. Analysis Effectiveness

Based on the technical characteristics of our approach: (1) the privacy policy analysis effectiveness hinges on factors such as the choice of learning algorithms and the quality of corpus for training; (2) the code analysis effectiveness is deterministic based on the identified API/permission and the captured HTML files. We thus focus on the privacy policy analysis module in this research question, and discuss the limitations of code analysis in Section 6.

We first train a classifier for natural language sentences using our corpus PrivAud-100. To evaluate the accuracy and the robustness of our trained classifier, we utilize the existing corpora constructed for privacy policy analysis among mobile applications in the literature, Liu2021 [24] and APP-350 [51]. This is based on the consideration that the privacy policies from the same service provider usually remain the same for applications across various platforms such as those for mobile applications and browser extensions.

Experimental Settings. To select the most suitable classifier with the best sentence annotation performance, we apply three algorithms commonly used for the NLP tasks, including SVM, BiLSTM, and BERT. More specifically, we use the n-gram and TD-IDF features adopted in the literature [24] for SVM, with TD-IDF values calculated based on the training data in the PrivAud-100 corpus. We adopt the GloVe technique [29] to obtain the word vectors for training and testing in BiLSTM. We utilize the last layer vectorial output as the sentence representation in BERT. For parameter fine-tuning, we adopt the linear kernel function and the default settings for SVM. We follow the standard process [9, 15] to fine-tune the BiLSTM and BERT parameters. The target parameters, their search space and final values are listed in Table 7.

Table 7: Parameter Search Ranges and Final Values

Parameters	Search Ranges	Final Tuned Values
Max Input Data Dimension	[32,64,128,256,512,768, 1024]	768(BERT), 256(BiLSTM)
Optimizer	[Adam, Admax, RMSprop]	Adam
Learning Rate	[3E-5, 3E-4, 5E-4, 1E-4, 5E-3]	5E-3(BERT), 5E-4(BiLSTM)
Training Epochs	[10, ..., 100]	40(BERT), 100(BiLSTM)
Batch Size	[4, 8, 16, 32, 64,128]	8(BERT), 64(BiLSTM)
Dropout	[0.1, ..., 0.9]	0.1(BERT), 0.5(BiLSTM)

When evaluating the classification performance of the two corpora, we test our trained models on them separately based on the similarity of the classification tasks. As mentioned in Section 3.3, we first evaluate the notification type classification for sentences in Liu2021, followed by the data collection type classification for sentences in APP-350. To benchmark the performance in both steps, we compare the results with that derived from our models, as shown in Table 8. For an unbiased evaluation, we apply 10-fold cross-validation on the classifiers. Each corpus is split into 10 folds, with 8 for training, 1 for parameter tuning and optimization, and 1 for testing. To further reflect the actual performance of each model, we calculate the average performance over 5 evaluation cycles, measured by commonly-used metrics including precision, recall, and F-1 score.

Results. Considering the relatively stable performance across different labels, we only show the weighted average results for simple and direct comparison, despite the dominance of the sentences without target information (labeled as *Other Info* in the corpora). From the results, we observe that the BERT model trained on PrivAud-100 corpus yields the best test results across all three corpora, indicating its high robustness and effectiveness in annotating unseen policy sentences. Similar to the observations made in the related works [24, 51], the ambiguity in the natural language sentences results in relatively lower recall and F1 scores compared to precision.

Quality Analysis on Results. Given the goal of facilitating the privacy compliance violation diagnosis, our approach with the achieved classification performance (over 90%) is sufficient in effectively narrowing down the search space for possible violations. Meanwhile, we note that manual efforts are still required to eliminate the false positives and confirm the actual violations. To obtain insights on causes that hinder a higher accuracy, we further examine the misclassified sentences. We identify two major sources of errors. One is related to poor negation detection where the model confuses negated and affirmed sentences. For example, the model labels the sentence "`We do not collect personally`

Table 8: Trained Classifier Performance on Three Corpora

Algo	Corpus	Precision	Recall	F1
SVM	Liu2021	0.717	0.741	0.725
	APP-350	0.674	0.608	0.639
	PrivAud-100	0.876	**0.810**	0.836
BiLSTM	Liu2021	0.710	0.565	0.629
	APP-350	0.494	0.466	0.479
	PrivAud-100	0.752	0.753	0.745
BERT	Liu2021	0.884	0.797	**0.838**
	APP-350	0.854	0.698	0.768
	PrivAud-100	**0.910**	0.700	0.785

`identifiable data (such as your name, email address, etc.)`" (with the label *Other Info*) as **DC**. The other major source is related to poor semantics recognition where the model misclassifies an actually irrelevant sentence containing target keywords as relevant. For example, the model labels the sentence "`Cookies are files with small amount of data which may include an anonymous unique identifier.`" (with the label *Other Info*) as **AI**. We will further discuss this in Section 6.

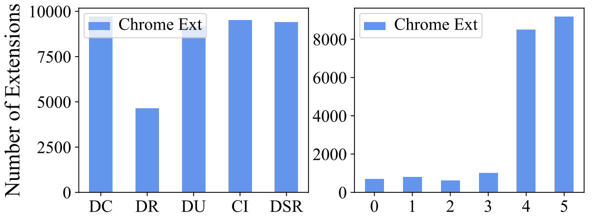
> **Answer to RQ1:** The trained BERT model has shown the best accuracy and robustness in annotating the privacy policy texts, achieving a similar performance compared to the state-of-the-art approaches. We will thus employ this model to annotate the rest privacy policies from our dataset.

5.2 RQ2. State of Privacy Compliance

In this section, we systematically conduct the diagnostic privacy compliance evaluation for extensions in our dataset. We focus on the completeness of the user notification information in the privacy policy, and the consistency between the privacy policy/PPS and the actual extension practices.

5.2.1 State of Privacy Policy Compliance. The absence of a valid privacy policy page/URL is directly deemed to be incompliant with the privacy regulations. We thus filter the extensions in our dataset by policy availability. To our great surprise, the vast majority of them (i.e., 43,353 out of 64,114, or 67.6%) do not have one.

Among the remaining 20,761 extensions, we employ our classifier for the privacy policy text annotation. Based on the results, we count the number of policies that present a particular category of notification information, as shown in Figure 6(a). It is observed that the data retention period is prone to be ignored by the service providers, as only 44.7% of the policies include this information compared with at least 88.3% for the other four categories. This could indicate the prevalent excessive user data collection, especially prolonged or even indefinite data storage. To obtain an overview of the privacy policy compliance state, we list the number of notification information categories each extension contains, as shown in Figure 6(b). Note that there are 702 extensions with privacy policies written in other languages, we follow the literature and mark them as containing zero labels. From the result, we find only 44.1% (of 20,761) covers the complete notification categories, reflecting the poor compliance of extension privacy policies versus GDPR.



(a) Notification Category (b) No. of Notification Category

Figure 6: Notification Compliance by Notification Information Category and Number of Notification Information Categories Covered in A privacy Policy

5.2.2 State of Privacy Practice Compliance. Despite having a privacy policy that is compliant with regulations, an extension's actual behaviors during runtime could still significantly deviate from the policy declarations. This is mostly manifested in the data collection practices, as discussed in Section 3.1. In this research question, we target the following two aspects of data collection practice consistency: 1) does an extension actually collect personal information; 2) if it does, are the types of collected information the same as those declared in the privacy policy?

We follow the method described in Section 3.3 to extract the collection behaviors regarding the three types of user data from API usage (listed in Table 4) and HTML files (listed in Table 5). By comparing them with the set of labels annotated from the privacy policy, we summarize the practice compliance status with respect to each data type in Table 9. We further analyze the practice compliance versus the notification information completeness in the privacy policy, as shown in Figure 7. In particular, the status "Over declare" means an extension declares to collect this type of user data in its privacy policy but never does during runtime, and vice versa. The status "Null" indicates the absence of both privacy policy and data collection behavior for the data type.

Practice Compliance versus Privacy Policy. On a finer-grained view in Figure 7, we observe no significant deviations in practice compliance according to the number of notification information categories in the privacy policy. This suggests that the practice compliance is independent of the privacy policy compliance. On a broader view as shown in Table 9, we have identified a prevalent inconsistency between the declared and the actual data collection practices, with only 8.5% to 15.4% of the extension correctly following their promises across the three data types. Meanwhile, extensions tend to hide their data-related actions by declaring fewer types of data accessed by them. For example, nearly 50% of the extensions collect excessive device information. Similarly, around a quarter secretly collect more personal information. It is also interesting to notice some extensions over-declare their data collection practices. For example, as many as 23.5% of the extensions falsely claim that they collect user activity information. This is likely due to the deficiency of privacy expertise among the developers.

Practice Compliance versus PPS. We reveal that the privacy-preserving measures enforced by the Chrome Extension Store are plausibly insufficient, as we observe a significant level of inconsistency between PPS and data collection practices. As shown in Table 9, as low as 37% of the developers correctly declare the device information that they collect. This reveals the absence of an efficient

Table 9: Practice Compliance versus Data Types

Practice Compliance Status		**PI**(Personal Info)	**AI**(Activity Info)	**DI**(Device Info)
Privacy policy	Over declare	9261	15097	1061
	Correct declare	**9884**	**5443**	**9144**
	Under declare	15939	3589	30090
	Null	29030	39985	23819
PPS	Over declare	1159	2775	0
	Correct declare	**33162**	**44044**	**23803**
	Under declare	20996	6354	35722
	Null	8797	10941	4589
Total		64114	64114	64114

diagnostic mechanism to validate the declared information from the developers versus the runtime characteristics of the extensions.

Validity Analysis of Practice Compliance Results. Considering that errors could be introduced during both privacy declaration and practice analysis, we further conduct a validity analysis on the reported results. To this end, we assemble two validation datasets 𝕍1 and 𝕍2 for privacy policy and PPS respectively, containing 50 randomly sampled extensions from over, correctly and under declared categories (i.e., each dataset contains 150 extensions). We conduct a manual examination of all the extensions in the two datasets to identify incorrectly reported cases. Considering the similar nature between the three information labels, we simply use **PI** for demonstration.

- For privacy policy analysis, the results are related to the language model annotation accuracy. We observe an overall accuracy of 90.3%, with 14 (8 and 6 in the over and correctly declared categories respectively) false positives and 2 (correctly declared category) false negatives. The error comes from the wrong handling of negated sentiment, as mentioned in Section 5.1. The higher occurrence of prediction error is related to the relatively longer policy texts among over-declared extensions, and vice versa.
- For PPS analysis, we found no false positives/negatives as the mapping from the data types to labels is deterministic.
- For practice analysis, the only possible cause of false negatives is dynamic analysis (i.e., deterministic but incomplete). We find that all the extensions in 𝕍1 and 𝕍2 will collect the user information from shallow layers of UIs (i.e., depth less than 2), as detailed in Section 3.3.4. Therefore, our dynamic analysis will capture the data collection practices if any. For extensions in general, we also study the UI complexity to confirm the effectiveness of our dynamic analysis. Among the 64,114 extensions, only 18,915 extensions have UIs, out of which 3,243 have deep layer interfaces (i.e., depth 3 or more). By randomly sampling 30 extensions from them, we manually confirm none of them collect user data from deep layers. Therefore, our dynamic analysis retains high effectiveness.

Answer to RQ2: Through our analysis, we observe the low compliance in both privacy policy and actual practices among extensions. Overall, there are 14.3% of extensions fully compliant with the notification completeness in the privacy policy, and 8.5% of extensions compliant with the declared data collection practices. We also reveal the ineffectiveness of the PPS enforced by the Chrome Extension Store, given its significant level of inconsistency against the actual practices.

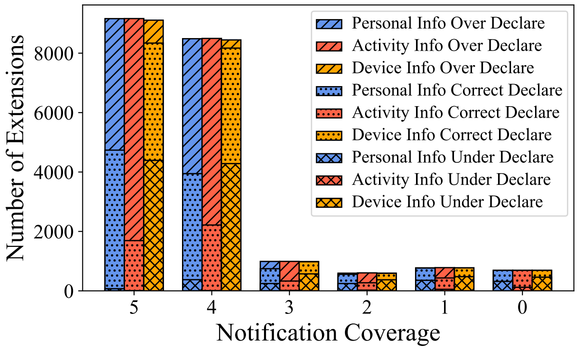


Figure 7: Practice Compliance vs. Policy Notification

5.3 RQ3.Characteristics of Extension Privacy (In)Compliance

To facilitate the implementation of a diagnostic system for extension privacy compliance, we seek to identify and examine the policy-based and code-based features that can link an extension to privacy compliance/incompliance. Based on them, we further propose a privacy compliance index to assist the understanding and interpretation of the relevant features. Using this index as a simplified and complementary version of our privacy compliance violation detection approach, the extension users, developers and store operators can quickly flag suspicious ones without an in-depth analysis.

5.3.1 Feature Relevance. We first assemble a benchmark set of extensions, and then discuss the relevance of each feature.

Benchmark Extension Sets. We randomly sample 300 extensions with fully compliant privacy policies and practices from our evaluation in RQ2. As a comparison group, we also randomly sample 300 extensions that are only partially compliant in the two aspects.

Evaluating the Feature Relevance. We aim to identify features that are indicative of the high-quality development lifecycle of an extension, which may further correlate with better privacy compliance. For this purpose, we propose 7 features spanning four aspects, including metadata of user-oriented statistics, policy quality, functional complexity and practice declarations, as shown in Table 10. Considering the quantitative nature of the features, we first determine their thresholds that yield the highest relevance to privacy compliance. For this purpose, we calculate the Odds Ratio [13] of each feature between two benchmark extension sets using various threshold values, and adopt those with the highest Odds Ratio values. Note that a value greater than 1 indicates that the feature is positively correlated to privacy compliance, and a value between 0 and 1 indicates otherwise. Then, we compare the feature relevance, as shown in Table 10.

We observe distinct patterns for the features relevant to policy and practice compliance respectively. As shown in Table 10, almost all features are positively correlated to privacy compliance, except the *number of UI button* feature for practice compliance (Odds Ratio=0.795). The policy compliance is most relevant to the policy length (Odd Ratio=109.4) where a longer text has a higher probability of comprehensively covering the required notification information. In comparison, practice compliance has stronger relevance to the user-based and code-based features, such as the number of requested permissions from the source code (Odds Ratio=30.25)

Table 10: Odd Ratio Values of Extension Characteristics

Compliance Type	Feature Summary						
	Extension Metadata		Policy	Extension Source Code			Platform (PPS)
	Download No.>10,000	Rating Score>4.5	Sentence No.>75	Permission No.>2	API No.>20	UI Buttons No.>10	Practice No.>3
Policy Compliance	1.030	1.000	**109.4**	1.927	1.224	1.104	2.514
Practice Compliance	2.119	2.130	2.283	**30.25**	1.944	0.795	**17.17**

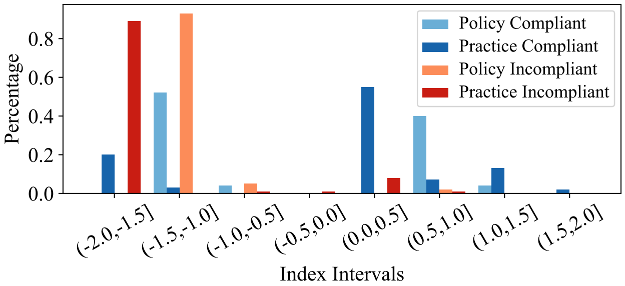


Figure 8: Compliance Index Distribution

and the number of declared data-related practices on PPS (Odds Ratio=17.17). Intuitively, the more permissions requested or practices declared by an extension, the better its developer understands the runtime practices. This will further render it less likely to conduct excessive data collection behavior inadvertently.

5.3.2 Compliance Index. Based on the feature relevance information, we further propose an index to measure the likelihood of an extension being privacy compliant/incompliant in its policy and practices respectively. To do so, we assign an index value to each feature listed in Table 10, which is given by its normalized Odds Ratio value. For example, if a privacy policy has more than 75 sentences, we assign the normalized values 1 (i.e., calculated by 109.4/109.4) and 0.075 (i.e., calculated by 2.283/30.25) to its policy and practice compliance indexes respectively. In the other case, if the feature is not satisfied (i.e., a privacy policy has less than or equal to 75 sentences), we will assign the negative normalized Odds Ratio values to the feature indexes (i.e., -1 and -0.075 respectively). The intuition is to place heavier penalties on features that have a higher correlation to privacy compliance. The overall compliance index value is derived by the sum of all feature index values.

To further determine its effectiveness in gauging privacy compliance, we randomly select another dataset containing 400 non-overlapping extensions included in the benchmark extension set for an unbiased evaluation. Among the 400 extensions, we have 100 policy compliant/incompliant extensions separately, and 100 practice complaint/incompliant extensions separately. According to our definition, the index value range is [-14,14]. However, we have found all the compliance indexes of the 400 extensions fall in the range of [-2,2], as shown in Figure 8. We observe that a negative index value is strongly indicating privacy incompliance, with over 90% accuracy. To confirm its effectiveness, we randomly selected 40 extensions with negative index values, and identify 37 (92.5%) of them are privacy incompliant. Meanwhile, a positive index is less indicative as the simple index is less capable of reflecting the satisfaction of a set of sophisticated requirements.

Answer to RQ3: We have identified three features highly relevant to privacy compliance, including the number of sentences in a privacy policy (policy compliance), the number of requested permissions (practice compliance) and the number of declared privacy practices (practice compliance). Based on the features, we have proposed a privacy compliance index that shows over 90% accuracy for identifying incompliant extensions.

6 DISCUSSIONS

6.1 Implications

Our findings reveal the prevalent weak/partial privacy compliance in the browser extension ecosystem, which is further linked to the user's increasingly-concerned issue of personal data abuse. This privacy compliance insufficiency sheds light on the necessity to implement an end-to-end diagnostic system proposed in this work.

First, effectively understanding the privacy regulations written in natural language has been a barrier for average users without law-related expertise. It is thus desirable to construct an NLP model to facilitate the privacy policy interpretation, which is capable of automatically checking the completeness of the information required by the privacy regulations and articulating the data collected/accessed by the service providers.

Second, the lack of privacy-specific domain knowledge among developers could easily lead to extension practices significantly deviating from those declared in the privacy policy. Considering such deviations commonly involve data handling without user consent, it is imperative for the extension platforms to develop tools for effectively detecting such inconsistencies. It is also critical to design and implement privacy vetting processes, instead of simply relying on declarations made by developers.

6.2 Limitations and Threats to Validity

To the best of our knowledge, this is the first work that systematically studies the status quo of the privacy compliance of browser extensions at scale. However, there are a few threats to validity that should be further investigated in future works.

Threats to Internal Validity. First, the automatic privacy policy annotation using NLP models inevitably causes inaccuracies, similar to other image-based models [46, 47]. As noted in Section 5.1, we have identified two main types of misclassifications. Considering our approach aims to warn the extension developers and store operators of the possible privacy compliance violations, a manual inspection is still required to confirm the actual violations. Second, the dynamic analysis is incomplete. For efficiency, we only traverse through the first two layers of an extension's UIs. This strategy excludes the HTML files that might be generated by the more sophisticated deeper-layer elements during runtime. Considering the relatively simple structure of the extensions, our method still retains fidelity as most of the elements related to user data collection/access are located on the superficial layer we target. This is confirmed by our validity analysis in Section 5.2.2.

Threats to External Validity. First, our current approach is only directly applicable to the requirements of GDPR. However, we also note its potential to be adapted for checking the privacy compliance requirements from other regulations such as CCPA and LGPD. Specifically, a regulation-specific corpus should be created and the personal data labels should be calibrated accordingly. The methodology for conducting policy and practice compliance analysis can be reused. Second, our method is limited to checking the practice compliance locally based on the extension source code (e.g., data collection). Analyzing the data management practices (e.g., data retention, data subject rights) on the service provider side remains challenging due to the lack of a verifiable framework in the current extension ecosystem. For example, a user can request a service provider to delete his/her profile. However, this user is unable to verify if all the stored information is indeed deleted on the server end. For future work toward a more privacy-compliant environment, a framework similar to a Single Sign-on scheme is desirable where the user is capable of revoking and modifying the previous data consent to a trusted data management entity.

7 RELATED WORK

In this section, we review the efforts in the literature towards analyzing the compliance of privacy policies and privacy practices from user-oriented applications. Our work is part of the efforts towards fostering the secure- and privacy-preserving environment for application users, complementing previous efforts [25, 42, 43] in the literature.

7.1 Analysis of Privacy Policy

The related analyses aim to propose approaches to facilitate the automatic annotation and quality evaluation of privacy policies (i.e., the information comprehensiveness versus privacy regulations).

Privacy Policy Corpora. Several corpora have been constructed for the automatic annotation task [19, 27, 32, 36, 45]. More recently, Zimmeck et al. [51] construct the APP-350 corpus targeting the privacy policies of mobile applications. Liu et al. [24] create a corpus containing sentences from 304 policies to facilitate the detection of compliance issues from privacy policies against GDPR Article 13.

Privacy Policy Annotation. There are a few contributions towards automatic policy annotations [23, 26]. Harkous et al. [17] propose a framework named Polisis to automatically annotate the previously unseen privacy policies with both high-level and fine-grained details. Sarne et al. [31] propose a framework for modeling and annotating the topics of privacy policies using unsupervised learning techniques.

Privacy Policy Quality Evaluations. Several works have studied the privacy policy quality [35, 48]. More recently, Linden et al. [21] conduct a large-scale study to compare the quality of over 6k privacy policies, before and after the enforcement of GDPR. Liao et al. [20] investigate the privacy policy effectiveness of voice applications on two major platforms.

In our work, we construct the corpus PrivAud-100 to enable a comprehensive annotation of the privacy policy sentences, covering GDPR regulation requirements and fine-grained data types collected during runtime. We also compare with corpora [24, 51].

7.2 Analysis of Privacy Policy Compliance

Since the enforcement of privacy regulations over the past years, many works [3, 6, 18, 28, 30, 38, 39] have targeted compliance checking against them. Among existing works, Fan et al. [12] propose a similar system to systematically check the GDPR requirements and the corresponding data practices implemented by mobile health applications. In comparison, we target more general and comprehensive types of browser extensions, and wider aspects of privacy.

7.3 Analysis of Privacy Practice Compliance

In addition to policy compliance analysis, there are several studies focusing on the inconsistencies between applications' privacy practices and their privacy policies [2, 34, 44]. To mitigate the issue, Yu et al. [49] design and implement a system named AutoPPG to automatically generate the privacy policy of an application, which ensures high compliance between the application's actual practice and the generated policy. For a similar purpose, Zimmeck et al. [50] develop PrivacyFlash Pro to generate privacy policies for iOS applications with high reliability and usability.

Compared to the existing studies, our work considers more comprehensive practice compliance by considering both privacy policy and privacy regulations. We focus on browser extensions while most prior works focus on mobile applications.

8 CONCLUSION

In this work, we propose an end-to-end diagnostic approach for privacy compliance violations among browser extensions, checking the privacy policy versus GDPR requirements and the actual privacy practices. We utilize state-of-the-art language processing models to analyze the policy texts and a hybrid analysis to analyze the extension source code. To evaluate our approach, we have assembled a large-scale dataset containing 64,114 browser extensions, and have identified only 8% are fully privacy compliant. To the best of our knowledge, this work is the first comprehensive study on the status quo of privacy compliance among browser extensions. Our work should raise an alert to the extension users, service providers, and platform operators, and would encourage solutions toward better privacy compliance in the community.

ACKNOWLEDGMENTS

We thank the anonymous shepherd and reviewers for improving this manuscript. This research is supported by Singapore Ministry of Education Academic Research Fund Tier 3 under MOE's official grant number MOE2017-T3-1-007. This research is also supported by the University of Queensland under Global Strategy and Partnerships Seed Funding and the NSRSG grant 4018264-617225, National Key R&D Program of China (2021YFB2701000), the National Natural Science Foundation of China (grant No.62072046), and the Fundamental Research Funds for the Central Universities (HUST 3004129109).

REFERENCES

[1] Directive 95/46/EC. 1995. https://edps.europa.eu/data-protection/our-work/publications/legislation/directive-9546ec_en,visited in August 2022.

[2] Benjamin Andow, Samin Yaseer Mahmud, Justin Whitaker, William Enck, Bradley Reaves, Kapil Singh, and Serge Egelman. 2020. *Actions Speak Louder than Words: Entity-Sensitive Privacy Policy and Data Flow Analysis with POLICHECK.*

[3] Vanessa Ayala-Rivera and Liliana Pasquale. 2018. The Grace Period Has Ended: An Approach to Operationalize GDPR Requirements. In *RE*. 136–146.

[4] BeautifulSoup4. 2014. https://pypi.org/project/beautifulsoup4,visited in August 2022.

[5] Brazil. 2018. General Personal Data Protection Law. https://iapp.org/resources/article/brazilian-data-protection-law-lgpd-english-translation,visited in August 2022.

[6] Cheng Chang, Huaxin Li, Yichi Zhang, Suguo Du, Hui Cao, and Haojin Zhu. 2019. Automated and Personalized Privacy Policy Extraction Under GDPR Consideration. In *WASA*.

[7] Chrome Extension Dataset Repository. Visited in August 2022. https://github.com/ExtPPCompliance/PPCompliance.

[8] Corinna Cortes and Vladimir Vapnik. 1995. Support-Vector Networks. *Mach. Learn.* 20, 3 (sep 1995), 273–297.

[9] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *NAACL-HLT*. 4171–4186.

[10] Esprima. Visited in August 2022. https://esprima.org.

[11] InsertLearning Chrome Extension. 2017. https://chrome.google.com/webstore/detail/insertlearning/dehajjkfchegiinhcmoclkfbnmpgcahj,visited in August 2022.

[12] Ming Fan, Le Yu, Sen Chen, Hao Zhou, Xiapu Luo, Shuyue Li, Yang Liu, Jun Liu, and Ting Liu. 2020. An Empirical Evaluation of GDPR Compliance Violations in Android mHealth Apps. In *ISSRE*.

[13] Afina S. Glas, Jeroen G. Lijmer, Martin H. Prins, Gouke J. Bonsel, and Patrick M.M. Bossuyt. 2003. The diagnostic odds ratio: a single indicator of test performance. *Journal of Clinical Epidemiology* 56, 11 (2003), 1129–1135.

[14] Global Desktop Browser Market Share for 2022. Visited in August 2022. https://kinsta.com/browser-market-share.

[15] Alex Graves, Navdeep Jaitly, and Abdel-rahman Mohamed. 2013. Hybrid speech recognition with Deep Bidirectional LSTM. In *2013 IEEE Workshop on Automatic Speech Recognition and Understanding*. 273–278.

[16] Kevin A Hallgren. 2012. Computing inter-rater reliability for observational data: an overview and tutorial. *Tutorials in quantitative methods for psychology* 8, 1 (2012), 23.

[17] Hamza Harkous, Kassem Fawaz, Rémi Lebret, Florian Schaub, Kang G. Shin, and Karl Aberer. 2018. Polisis: Automated Analysis and Presentation of Privacy Policies Using Deep Learning. In *USENIX Security*. 531–548.

[18] Christos Karageorgiou Kaneen and Euripides G.M. Petrakis. 2020. Towards evaluating GDPR compliance in IoT applications. *KES* 176 (2020), 2989–2998.

[19] Logan Lebanoff and Fei Liu. 2018. Automatic Detection of Vague Words and Sentences in Privacy Policies. In *EMNLP*. 3508–3517.

[20] Song Liao, Christin Wilson, Long Cheng, Hongxin Hu, and Huixing Deng. 2020. Measuring the Effectiveness of Privacy Policies for Voice Assistant Applications. In *ACSAC*. 856–869.

[21] Thomas Linden, Hamza Harkous, and Kassem Fawaz. 2020. The Privacy Policy Landscape After the GDPR. *PETS* 2020 (2020), 47 – 64.

[22] List of Chrome extensions. Visited in August 2022. https://github.com/DebugBear/chrome-extension-list.

[23] Fei Liu, Rohan Ramanath, Norman M. Sadeh, and Noah A. Smith. 2014. A Step Towards Usable Privacy Policy: Automatic Alignment of Privacy Statements. In *COLING*.

[24] Shuang Liu, Baiyang Zhao, Renjie Guo, Guozhu Meng, Fan Zhang, and Meishan Zhang. 2021. Have You Been Properly Notified? Automatic Compliance Analysis of Privacy Policy Text with GDPR Article 13. In *WWW*. 2154–2164.

[25] Kulani Mahadewa, **Kailong Wang**, Guangdong Bai, Ling Shi, Yang Liu, Jin Song Dong, and Zhenkai Liang. 2019. Scrutinizing Implementations of Smart Home Integrations. *IEEE Transactions on Software Engineering (TSE)* (2019). https://doi.org/10.1109/TSE.2019.2960690

[26] Najmeh Mousavi Nejad, Simon Scerri, and Jens Lehmann. 2018. KnIGHT: Mapping Privacy Policies to GDPR. In *Knowledge Engineering and Knowledge Management*, Catherine Faron Zucker, Chiara Ghidini, Amedeo Napoli, and Yannick Toussaint (Eds.).

[27] Kanthashree Mysore Sathyendra, Shomir Wilson, Florian Schaub, Sebastian Zimmeck, and Norman Sadeh. 2017. Identifying the Provision of Choices in Privacy Policy Text. In *EMNLP*. 2774–2779.

[28] Monica Palmirani and Guido Governatori. 2018. Modelling Legal Knowledge for GDPR Compliance Checking. In *JURIX*, Vol. 313. 101–110.

[29] Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *EMNLP*. 1532–1543.

[30] Irwin Reyes, Primal Wijesekera, Joel Reardon, Amit Elazari, Abbas Razaghpanah, Narseo Vallina-Rodriguez, and Serge Egelman. 2018. “Won’t Somebody Think of the Children?” Examining COPPA Compliance at Scale. *PETS* (06 2018), 63–83.

[31] David Sarne, Jonathan Schler, Alon Singer, Ayelet Sela, and Ittai Bar Siman Tov. 2019. Unsupervised Topic Extraction from Privacy Policies. In *WWW*. 563–568.

[32] Kanthashree Mysore Sathyendra, Florian Schaub, Shomir Wilson, and Norman M. Sadeh. 2016. Automatic Extraction of Opt-Out Choices from Privacy Policies. In *AAAI*.

[33] Singapore. 2012. Personal Data Protection Act. https://www.pdpc.gov.sg/Overview-of-PDPA/The-Legislation/Personal-Data-Protection-Act,visited in August 2022.

[34] Rocky Slavin, Xiaoyin Wang, Mitra Bokaei Hosseini, James Hester, Ram Krishnan, Jaspreet Bhatia, Travis D. Breaux, and Jianwei Niu. 2016. Toward a Framework for Detecting Privacy Policy Violations in Android Application Code. In *ICSE*. 25–36.

[35] Peter Story, Sebastian Zimmeck, and Norman M. Sadeh. 2018. Which Apps Have Privacy Policies? - An Analysis of Over One Million Google Play Store Apps. In *APF*.

[36] Welderufael B. Tesfay, Peter Hofmann, Toru Nakamura, Shinsaku Kiyomoto, and Jetzabel Serna. 2018. I Read but Don’t Agree: Privacy Policy Benchmarking Using Machine Learning and the EU GDPR. In *WWW*. 163–166.

[37] The Selenium Project. Visited in August 2022. https://www.selenium.dev.

[38] Jake Tom, Eduard Sing, and Raimundas Matulevičius. 2018. Conceptual Representation of the GDPR: Model and Application Directions. In *BIR*, Jelena Zdravkovic, Jānis Grabis, Selmin Nurcan, and Janis Stirna (Eds.). 18–28.

[39] Damiano Torre, Ghanem Soltana, Mehrdad Sabetzadeh, Lionel Briand, Yuri Auffinger, and Peter Goes. 2019. Using Models to Enable Compliance Checking Against the GDPR: An Experience Report. In *MODELS*. 1–11.

[40] European Union. 2016. General Data Protection Regulation. https://gdpr-info.eu,visited in August 2022.

[41] California (USA). 2018. California Consumer Privacy Act. https://oag.ca.gov/privacy/ccpa,visited in August 2022.

[42] Kailong Wang, Junzhe Zhang, Guangdong Bai, Ryan Ko, and Jin Song Dong. 2021. It’s Not Just the Site, It’s the Contents: Intra-domain Fingerprinting Social Media Websites Through CDN Bursts. In *30th The Web Conference (WWW)*.

[43] Kailong Wang, Yuwei Zheng, Qing Zhang, Guangdong Bai, Qin Mingchuang, Donghui Zhang, and Jin Song Dong. 2022. Assessing Certificate Validation User Interfaces of WPA Supplicants. In *MobiCom*.

[44] Takuya Watanabe, Mitsuaki Akiyama, Tetsuya Sakai, and Tatsuya Mori. 2015. Understanding the Inconsistencies between Text Descriptions and the Use of Privacy-sensitive Resources of Mobile Apps. In *SOUPS*. 241–255.

[45] Shomir Wilson, Florian Schaub, Aswarth Abhilash Dara, Frederick Liu, Sushain Cherivirala, Pedro Giovanni Leon, Mads Schaarup Andersen, Sebastian Zimmeck, Kanthashree Mysore Sathyendra, N. Cameron Russell, Thomas B. Norton, Eduard Hovy, Joel Reidenberg, and Norman Sadeh. 2016. The Creation and Analysis of a Website Privacy Policy Corpus. In *ACL*. 1330–1340.

[46] Yan Xiao, Ivan Beschastnikh, Yun Lin, Rajdeep Singh Hundal, Xiaofei Xie, David S Rosenblum, and Jin Song Dong. 2022. Self-Checking Deep Neural Networks for Anomalies and Adversaries in Deployment. *IEEE Transactions on Dependable and Secure Computing* 01 (2022), 1–18.

[47] Yan Xiao, Ivan Beschastnikh, David S Rosenblum, Changsheng Sun, Sebastian Elbaum, Yun Lin, and Jin Song Dong. 2021. Self-Checking Deep Neural Networks in Deployment. In *2021 IEEE/ACM 43rd International Conference on Software Engineering (ICSE)*. IEEE, 372–384.

[48] Le Yu, Xiapu Luo, Xule Liu, and Tao Zhang. 2016. Can We Trust the Privacy Policies of Android Apps?. In *DSN*. 538–549.

[49] Le Yu, Tao Zhang, Xiapu Luo, Lei Xue, and Henry Chang. 2017. Toward Automatically Generating Privacy Policy for Android Apps. *TIFS* 12, 4 (2017), 865–880.

[50] Sebastian Zimmeck, Rafael Goldstein, and David Baraka. 2021. PrivacyFlash Pro: Automating Privacy Policy Generation for Mobile Apps. In *NDSS*.

[51] Sebastian Zimmeck, Peter Story, Daniel Smullen, Abhilasha Ravichander, Ziqi Wang, Joel Reidenberg, N. Cameron Russell, and Norman Sadeh. 2019. MAPS: Scaling Privacy Compliance Analysis to a Million Apps. *PETS* 3 (2019), 66–86.