Are Mobile Advertisements in Compliance with App’s Age Group?

Yanjie Zhao∗
School of Software, Beihang University
China

Tianming Liu†
Faculty of Information Technology,
Monash University
Australia

Haoyu Wang
Huazhong University of Science and Technology
China

Yepang Liu
Southern University of Science and Technology
China

John Grundy
Faculty of Information Technology,
Monash University
Australia

Li Li‡
School of Software, Beihang University
China

ABSTRACT
As smartphones and mobile apps permeate every aspect of people’s lives, children are accessing mobile devices at an increasingly younger age. The inescapable exposure of advertisements in mobile apps to children has grown alarmingly. Mobile advertisements are placed by advertisers and subsequently distributed by ad SDKs, under the rare control of app developers and app markets’ content ratings. Indeed, content that is objectionable and harmful to children’s mental health has been reported to appear in advertising, such as pornography. However, few studies have yet concentrated on automatically and comprehensively identifying such kid-unsuitable mobile advertising. In this paper, we first characterize the regulations for mobile ads relating to children. We then propose our novel automated dynamic analysis framework, named AdRambler, that attempts to collect ad content throughout the lifespan of mobile ads and identify their inappropriateness for child app users. Using AdRambler, we conduct a large-scale (25,000 mobile apps) empirical investigation and reveal the non- incidental presence of inappropriate ads in apps with child-included target audiences. We collected 11,270 ad views and identified 1,289 ad violations (from 775 apps) of child user regulations, with roughly half of the app promotions not in compliance with host apps’ content ratings. Our empirical investigation and reveal the non-incidental presence of inappropriate ads in apps with child-included target audiences. We further delve into the question of accountability for the presence of inappropriate advertising and provide concrete suggestions for all stakeholders to take action for the benefit of children.

CCS CONCEPTS
• Security and privacy → Software and application security; • Information systems → Online advertising; • Human-centered computing → Ubiquitous and mobile computing.

KEYWORDS
Mobile advertising, Android app, Malvertising

ACM Reference Format:

1 INTRODUCTION
Between 2017 and 2022, there has been a massive boom in smartphone users, with a 49.89% increase in the number of people owning a smart cell phone. Statistically, the number of smartphone users globally as of 2022 is 6.6 billion, corresponding to 83.37% of the world’s population using a smartphone [39]. As the mobile industry develops, mobile devices have penetrated every aspect of people’s lives, with no exception for children. According to a survey by the Pew Research Center [8], 60% of parents with a kid under the age of 12 indicate that their child has ever used or interacted with a smartphone, and 31% in that group are exposed to smartphones before age 5. Thus an increasing number of children are using smartphones at an increasingly younger age.

Given that mobile devices have gained such popularity among children, the public and the research community have commenced raising awareness about children being exposed to inappropriate information in recent years, especially after the notorious Elsagate [42] controversy emerged in late 2017. Thousands of videos are found on YouTube Kids and other video platforms that portray themselves as “child-friendly”, featured by beloved cartoon characters such as “Elsa” from Disney animated film Frozen [43]. However, these videos involve inappropriate themes for children, such as sexual situations, graphic violence, fetishes, and drugs.

Mobile advertising fuels the economy of much of the mobile app ecosystem. According to traditional estimates, children begin to comprehend the persuasive intent of advertising around the age of 8 [15]. As advertisements (hereinafter referred to as “ad” or “ads”) in mobile apps are generally served by third-party ad networks, the developers and app markets are in less control over the ads presented within the apps than the actual content of the apps. The inappropriateness of advertising within apps with child-friendly content ratings hence deserves significant attention. Indeed, the appearance of inappropriate ads with sexually-suggestive content or even pornography in children’s apps from Google Play has been recurrently reported over the years [29, 36, 37]. Yet despite so many complaints and Google Play’s phased update on child protection policies [4], inappropriate advertising still continues to appear. Chen et al. [10] performed a manual analysis on the inappropriateness of the in-app advertising in 405 mobile apps designed...
for children in 2013. Follow-up researches [9, 27] have done some preliminary examinations and analyses but are limited in scope. Few studies to date have concentrated on automatically detecting children’s inappropriate exposure to mobile app advertising.

To fill this gap, we propose a prototype framework named AdRambler for automatically and precisely clicking the ads and collecting ad-related data, primarily based on dynamic analysis techniques. AdRambler aids in automatically triggering numerous ads on a vast number of apps whose target audience includes children to promote further analysis regarding whether these apps contain inappropriate ads. Our evaluation results demonstrate that AdRambler can detect and record ads precisely, and that there are, unfortunately, plenty of mobile app ads that violate the relevant policies for children’s protection. We further discuss who share the blame for the presence of inappropriate advertising and provide tangible recommendations for all relevant parties to take action in this matter.

We make the following key contributions in this work:

• We propose a prototype tool, named AdRambler, for automatic dynamic analysis of Android apps to precisely click the ads and collect ad-related data required for further identification of ad inappropriateness;
• We conduct a large-scale empirical study in the wild with AdRambler to capture and record app ads and identify the presence of violations for advertising in realistic apps. By applying AdRambler to 25,000 Google Play apps, we collected 11,270 ad views and found that 775 apps delivered 1,289 ad violations for child users; and
• We delve into the question of accountability for the presence of inappropriate advertising and offer concrete suggestions for all stakeholders to take action for children’s sake.

2 BACKGROUND

2.1 Motivation

In our preliminary research, we have noticed that many apps with child-friendly content ratings contain inappropriate ads. As a concrete example, consider Figure 1, which displays the triggering process in an ad included in the App Lion Roar Sound, which provides lion roar sounds for users. It is easy to observe from Figure 1(a) that the app’s audiences are EVERYONE, of which children are not excluded. When we launch the app, however, it pops up a video-form ad, as shown in Figure 1(b). This ad eventually induces the user to download a dating app suitable for ages 17 and up, i.e., App Bumble illustrated in Figure 1(c), which is an obvious violation of user guideline regulations.

This motivating example suggests that apps that appear to be child-friendly on the surface could be potentially harmful, e.g., by directing children to premature exposure to inappropriate content with ads as a carrier. Therefore, we wanted to investigate relevant policies and detect such violating ads to warn and inspire the industry and research community.

2.2 Ad Inappropriateness for Children

Multiple laws and regulations are issued in different countries to safeguard children from the harm caused by the inappropriateness of mobile apps, such as the US Children’s Online Privacy and Protection Act (COPPA) [11] and the EU General Data Protection Regulation (GDPR) [18]. In compliance with these regulations, Google Play has assigned each app to an age-appropriate group. The content ratings and descriptions of age groups vary in different regions to comply with the local regulations [31]. For instance, the age appropriate groups in the North & South America are divided into EVERYONE, EVERYONE 10+, TEEN (13+), MATURE (17+), and ADULTS ONLY (18+). With the assigned age group, the app market could better maintain the age appropriateness of each app and reduce children’s exposure to improper apps.

It has been previously mentioned that the app market has difficulty regulating in-app ads as they are provided by third-party components. As such, Google Play has imposed a series of requirements on in-app advertising that developers must capitulate to if the target audience of their apps includes children [20]. The requirements can be summarized into two categories:

---

1AdRambler can be accessed via https://github.com/Tmliu06/AdRambler.

2Although different, they all classify people under the age of 12 or 13 as children, following local regulations [2, 12, 17]. Unless stated otherwise, we refer to children as anyone under 12 in this paper.
Ad SDK. For apps whose target audiences include only children and who serve ads using an ad SDK, the in-app ads can only be served by Google Play-certified ad SDKs [22], which currently include 11 well-known SDKs like Google AdMob, as listed in Table 1. These self-certified ad SDKs claim to rate their served ads according to age-appropriate groups and allow developers to request child-appropriate ads on a per-request or per-app basis. For apps targeting both children and older users, the developer must implement age screening measures to avoid serving ads to children from non-certified SDKs [3].

Table 1: The Google Play-certified Ad SDKs.

<table>
<thead>
<tr>
<th>Name</th>
<th>Example Package Name</th>
<th>Name</th>
<th>Example Package Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>Google AdMob</td>
<td>com.google.android.gms.ads</td>
<td>AdMob</td>
<td>com.admob</td>
</tr>
<tr>
<td>Google Ad Manager</td>
<td>com.google.google.af</td>
<td>AppLovin</td>
<td>com.applovin</td>
</tr>
<tr>
<td>Chartboost</td>
<td>com.chartboost</td>
<td>ironSource</td>
<td>com.ironsource</td>
</tr>
<tr>
<td>tv.superawesome</td>
<td>tv.superawesome</td>
<td>Unity Ads</td>
<td>com.unity3d.services.ads</td>
</tr>
<tr>
<td>Vungle</td>
<td>com.vungle</td>
<td>Total</td>
<td>11</td>
</tr>
</tbody>
</table>

Ad Content. Several types of explicit content are regarded as inappropriate [21]. They should not be displayed to children in apps as stated by Google Play, including pornography and sexually suggestive content, simulated or real gambling (even if free to enter), dating or adult relationships, controlled or harmful substances (e.g., alcoholic beverages and tobacco products), violent content, and any other inappropriate media content. Specifically, promoting inappropriate downloadable software (e.g., apps) to children is also disallowed.

In order to adequately detect these two categories of violations in mobile advertising, it is necessary to first capture and record as much ad-related content within apps as possible. Especially when dealing with large-scale in-the-wild apps, it is essential to automate the ad sniffing, triggering, and gathering.

3 OUR APPROACH: ADRAMBLER

This research aims to characterize and detect inappropriate ads in mobile apps whose audiences include children. We expect to expose the current presence of in-app advertising violations toward child protection policies. To this end, we propose an automatic framework called ADRambler to explore the apps and harvest ad-related information for further investigations. Figure 2 demonstrates the workflow of our framework, which is composed of three modules: (1) AdBot uses app automating techniques to traverse an app’s GUI pages and interact with in-app ads. To precisely identify and click the ad views during the exploration, AdBot captures the instantiating class for each GUI widget at runtime to map the GUI widgets with ad SDKs. (2) AdTraffic is responsible for the identification and collection of all the ad-related traffic during the exploration, which runs in parallel with AdBot. Taking advantage of the Socket Hooking approach, AdTraffic can recognize ad traffic that originates from ad SDKs. (3) AdGuard identifies any inappropriateness within the in-app ads based on pre-defined rules. Utilizing the recorded ad SDKs from AdBot and the ad-related content from AdTraffic, AdGuard can comprehensively characterize and inspect the ads’ inappropriateness towards children within apps.

3.1 Automated Exploration of In-app Ads

Serving as the first step of ADRambler, AdBot is designed as a dynamic exploration tool to interact with the in-app ads automatically.
Moreover, applying heuristic rules could also result in the omission of ad views, leading to false negatives in the inappropriateness detection. To ensure the comprehensiveness of our testing and to improve efficiency, we thus propose a novel ad view identification approach that utilizes framework hooking to link the ad view with its ad SDK.

**Identification of Ad Views.** The in-app ads are distributed by third-party ad SDKs embedded in the app. Different from normal non-ad views that are implemented by app developers or UI libraries [7], ad views are instantiated by specific View classes within the ad SDKs [16] (e.g., com.applovin.adview.AppLovinAdView, where com.applovin is the package name of an ad SDK). Therefore, by retrieving the class that instantiates the view, we can easily determine whether a view is an ad view. The implementation of our identification approach is detailed below. Recall that AdBot utilizes Accessibility Services to obtain the layout of each GUI page, we thus leverage framework hooking on Android Accessibility APIs, including View.onInitializeAccessibilityNodeInfo() and View.createAccessibilityNodeInfo(), to obtain the real class that instantiates the view. We then use a whitelist-based approach to identify ad views with an ad SDK list from a most recent work [5], which contains a total of 64 well-known ad SDKs and their corresponding package names. If a view’s instantiating class comes from an ad SDK’s package, we recognize the view as an ad view from that SDK and configure AdBot’s next input to click the ad view.

### 3.2 Identifying Ad-related Traffic

The second module of AdRambler, AdTraffic, is designed to harvest and identify all the ad-related network traffic at runtime, which runs in parallel with AdBot. To handle and decrypt HTTPS messages, AdTraffic implements a Man-in-The-Middle (MiTM) proxy service on the test device, where the proxy is configured to record the whole traffic from the test device in detail in a timely manner.

**Connecting Ad Traffic with Ad SDKs.** While an ad is being loaded and clicked during the exploration, network traffic unrelated to ads will also be captured inevitably (e.g., the app communicating with its own server). The key difference between ad traffic and non-ad traffic is that ad traffic is initiated by ad SDKs. We applied a ‘Socket Hooking’ approach to help identify ad traffic by connecting ad traffic with ad SDKs. The key idea is that any network communication in Android will commence with a socket system call [47]. We leverage framework hooking on socket APIs to monitor the network behavior on Android OS following the approach of previous studies [46, 47]. When a socket connection is established, our approach will automatically record the target host and dump the stack trace with the getStackTrace method. We can then locate the package initiating this connection. Recall that we have already obtained the ad SDK of ad views in AdBot. If the connection comes from the package of the ad SDK, we regard the connection as ad traffic.

**Collection of Ad Content.** The revenue of app developers from in-app advertising is generally based on either the number of ads displayed or the number of ads clicked, which are the two main phases of in-app advertising. Both phases will introduce ad content to users, which are called ad-loading content and ad-clicking content [26]. As shown in Figure 3, there are certain differences between the two types of ad content. First off, unlike ad-loading content that is delivered within the app, ad-clicking content often involves third-party apps such as Google Play promotions and web pages in a browser. In addition, ad-loading content will always be delivered to users’ devices even before the ad is displayed, while ad-clicking content may not be triggered unless the ad is clicked. Since both types of ad content have been stated to contain inappropriate artifacts to children in previous news reports [36, 37], AdTraffic aims to gather both types of ad content.

For the ad-loading content, we mainly focus on media files used in ad displays, such as images, videos, and web pages that construct the ad views, to detect any explicit content that may be inappropriate for children. In terms of ad-clicking content, two types are taken into account: (1) web pages: Upon clicking the ad, a series of redirection links will eventually lead the mobile device to a page displaying promoted ad information in a browser, which is collected for further detection. (2) app promotions: The host app is redirected to a Google Play page of the promoted app. We record the package name of the promoted app to detect further if it is suitable for children.

### 3.3 Identification of Inappropriate Ads

Finally, the third module of AdRambler, AdGuard, identifies the inappropriateness for children in ads harvested from mobile apps, using the output of AdBot and AdTraffic modules. We elaborate on the detection approaches for the inappropriateness of Ad SDK and Ad Content introduced in Section 2.2.

**Non-compliant Ad SDK Usage Detection.** As previously mentioned, Google Play explicitly disallows serving in-app ads to children through uncertified ad SDKs. Unlike certified SDKs which to a certain extent perform content rating procedures for their served ads, there is no guarantee that uncertified ad SDKs would follow the same patterns, rendering their underage audiences more susceptible to inappropriate or harmful advertising. Therefore, whether for the purpose of finding the culprit behind unsanitized behaviors, or to avoid potential hazards for child users, the usage of uncertified ad networks needs to be revealed. Recall that in the AdBot module, we have already obtained the ad SDK for each ad view, we then check the ad SDK against the Google Play-certified ad SDKs in Table 1 to expose the behavior of displaying ads through uncertified ad SDKs. Inappropriate Ad Content Identification. As mentioned in Section 3.2, we are interested in ad-loading content as well as ad-clicking content. The former needs to deal with media files such as the loading web pages, images (in JPEG, PNG, GIF, and WebP formats), and videos; for the latter, the clicking web pages, as well as the promoted apps, need to be considered. For the inspection of web pages in both phases, we resort to the Natural Language API\(^3\) provided by Google Cloud. According to the description of inappropriate ad content in Section 2.2, we collate 29 inappropriateness-related content categories\(^4\), such as /Games/Gambling, /Online Communities/Dating & Personals, etc. If a web page is determined to fall into one of these categories, its associated ad is considered harmful to children. For the inspection of images and videos, we apply Google Vision API\(^5\) to recognize the inappropriateness based on: (1) the Safe Search Detection feature, provided for explicit content.

\(^3\)https://cloud.google.com/natural-language
\(^4\)https://cloud.google.com/natural-language/docs/categories
\(^5\)https://cloud.google.com/vision
4 EVALUATION

Our investigations explore the experimental results with AdRambler to answer the following research questions:

- **RQ1**: How are mobile ads presented in real-world Android apps, and can AdRambler effectively recognize them?
- **RQ2**: Are there any policy violations within the in-app ads from apps for children audiences?
- **RQ3**: Does the inappropriateness of mobile advertising vary by region?
- **RQ4**: How well does AdRambler perform in ad collection compared with existing approaches?

All of our experiments are conducted on four parallel-running Nexus 5 smartphones. Recall that we mentioned in Section 2.2 that for apps with audiences including children and older users, age screening measures by developers are mandated to avoid serving inappropriate ads to children. In accordance with COPPA [13], Google Play demands that the age screening must be implemented in a neutral manner, requiring users to enter their birthdays freely, instead of presetting the birth date to a required age or simply providing a dialog to ask if the user is above a certain age [3]. As AdRambler is configured not to send any text inputs to the app, the age screening (if any) must assume the user is underage when serving ads. In other words, any ads displayed in our experiments can potentially be received by children.

4.1 Dataset

We adopt the widely-used AndroZoo dataset [6] to crawl Android apps from the official Google Play store for the experiments. Since we are only interested in apps that might display ads to children, apps with age-appropriate groups of EVERYONE or EVERYONE 10+ on Google Play are considered, as both groups include target audiences under 12 years old. Google Play marks each app as ad-containing or not; in this work, we only consider apps that contain ads in recent years. Hence, an app will be collected as a candidate only if it is (1) available on Google Play, (2) with an ad-containing tag marked by Google Play, (3) with an update time from 2018 to 2022, according to AndroZoo, and (4) is assigned with an age group that includes children. Eventually, we randomly selected 25,000 apps that met the above criteria to form our overall dataset.

4.2 RQ1: AdRambler Effectiveness

Our first research question concerns the effectiveness of our AdRambler framework for detecting ads in mobile apps. We conduct a large-scale study with AdRambler on our dataset composed of 25,000 in-the-wild Android apps. The investigation for this RQ is conducted in the Hong Kong network environment. Given that it takes time for in-app ads to be loaded from the ad networks, a time interval is set between the generated test inputs. Overall, an app is evaluated in 2 minutes on average.

| Category | Type          | # All | # Valid | # Kid-unsuitable
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Ad View</td>
<td>-</td>
<td>11,270</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Ad SDK</td>
<td>-</td>
<td>12</td>
<td>7/396</td>
<td>7/396</td>
</tr>
<tr>
<td>Ad-loading Content</td>
<td>Web Pages Images Videos</td>
<td>51,812</td>
<td>5,342</td>
<td>94/61</td>
</tr>
<tr>
<td></td>
<td>Ad-clicking Content</td>
<td>Web Pages App Promotions</td>
<td>53,461</td>
<td>12,187</td>
</tr>
<tr>
<td>In Total</td>
<td>-</td>
<td>173,295</td>
<td>43,058</td>
<td>1,289/775</td>
</tr>
</tbody>
</table>

As shown in Table 2, the presence of mobile advertising is widespread, as 11,270 ad views are identified out of 8,971 apps in our investigation. The median and mean numbers of ad views among all the 8,971 apps are 1 and 1.26. Most apps (7,031/8,971 = 78.37%) have one ad view, while 1,691 apps have two, and 249 apps have more than two.

Pre-filtering is required for all recorded ad-related content as there are many invalids. We filter the invalids in various ways for different types of content. Because many web pages contain very little semantic information and the Google Cloud Natural Language API has a minimum input length limit, we eliminate all web pages with less than 100 characters. In addition, we only preserve image files larger than 2KB in size, and video files that are not corrupted and can be opened normally. For the app promotions, since some apps have been unattainable in Google Play, promotions like these are regarded as invalid for detection. The statistics after filtering are shown in the fourth column of Table 2, where images are the most widely used advertising mediums. In the end, we collected 43,058 valid samples of various advertising.

Recall that we construct the dataset considering only apps labeled by Google Play as “Contains ads”. However, as shown in Table 2, we can only collect ads from 8,971 apps out of the 25,000 apps in our dataset. The main reasons are: (1) some apps require complex human interactions that cannot be simulated through general automation to reach the GUI pages that display ads, e.g., they request users to register and sign in before ad-showing; (2) some apps are inherently corrupted and will crash upon startup or after running only a few steps. Nevertheless, to the best of our knowledge, we achieve the highest percentage (8,971/25,000 = 35.88%) of in-app ads collected in similar state-of-the-art works, for instance, 21.51% for Liu et al. [26], and 3.99% for Chen et al. [9].

A total of 12 ad SDKs are used to display ads in these 8,971 apps, detailed in Table 3. Use of the ad SDKs provided by Google, i.e., Google AdMob and Google Ad Manager, lead the others by a
wide margin with a percentage of $7,894/9,046 = 87.27\%$. The next three most frequently used ad SDKs are StartApp, AppBrain, and AppLovin.

### Table 3: The presence of ad SDKs in Android apps.

<table>
<thead>
<tr>
<th>Ad SDK</th>
<th>Package Name</th>
<th>Frequency</th>
<th>Google Play Certified</th>
</tr>
</thead>
<tbody>
<tr>
<td>Google</td>
<td>com.google.android.gps.ads</td>
<td>7,894</td>
<td>✓</td>
</tr>
<tr>
<td>StartApp</td>
<td>com.startapp</td>
<td>995</td>
<td>✗</td>
</tr>
<tr>
<td>AppBrain</td>
<td>com.appbrain</td>
<td>290</td>
<td>✗</td>
</tr>
<tr>
<td>AppLovin</td>
<td>com.applovin</td>
<td>159</td>
<td>✓</td>
</tr>
<tr>
<td>Unity Ads</td>
<td>com.unity3d.services.ads</td>
<td>24</td>
<td>✓</td>
</tr>
<tr>
<td>Facebook</td>
<td>com.facebook.ads</td>
<td>24</td>
<td>✓</td>
</tr>
<tr>
<td>InMobi</td>
<td>com.inmobi</td>
<td>22</td>
<td>✓</td>
</tr>
<tr>
<td>Appnext</td>
<td>com.appnext</td>
<td>19</td>
<td>✗</td>
</tr>
<tr>
<td>MoPub</td>
<td>com.mopub</td>
<td>5</td>
<td>✓</td>
</tr>
<tr>
<td>Smaato</td>
<td>com.smaato</td>
<td>2</td>
<td>✓</td>
</tr>
<tr>
<td>IronSource</td>
<td>com.ironsource</td>
<td>2</td>
<td>✓</td>
</tr>
<tr>
<td>Applovin</td>
<td>com.applovin</td>
<td>5248</td>
<td>✓</td>
</tr>
<tr>
<td>Total</td>
<td></td>
<td>9,046*</td>
<td>✓</td>
</tr>
</tbody>
</table>

* One app may involve multiple ad SDKs.

### 4.3 RQ2: Detection of Inappropriate Ads

For our second research question, we aim to investigate the potential inappropriateness of in-app ads shown to child mobile app users. The last column of Table 2 lists the violations existing in our large-scale investigation. We depict the proportional percentage of the numbers of improper advertising in stacked bars, shown in Figure 4. We can conclude that **app promotion is the type of ad content most likely to be inappropriate**, with roughly 49.15\%, although it is not outstanding in quantity. In other words, approximately half of the promoted Google Play apps from host apps with child-included audiences are inappropriate for children. Video ad content also has a relatively higher percentage (25.65\%) to be kid-unsuitable. Although the proportion of unsafe ad images is not high, the number of images with inappropriate content is the most among all types of ad content. In Section 2 we introduced six categories of inappropriate ad content for child mobile app users. The numbers of inappropriate ad contents found in these categories are displayed in Figure 5, where the distribution of different advertising carriers is totaled. We can conclude that most of the kid-unsuitable content in ad-loading web pages is affiliated with simulated or real gambling, while in ad-clicking web pages, the inappropriateness is mainly associated with controlled or harmful substances (especially alcohol selling). For the inappropriate ad images, most of them (582/758 = 76.78\%) contain controlled or harmful substances, followed by pornography and sexually suggestive content (135/758 = 17.81\%). The inappropriate ad videos mainly consist of dating & adult relationships (103/177 = 58.19\%) and violence (41/177 = 23.16\%), e.g., the promotion of live video streaming platforms limited to 12+, games that contain violence inducement, and other mature-only apps.

![Figure 4: Proportion of various types of kid-unsuitable ad content.](image)

Since mobile ads are served through third-party ad SDKs integrated into the app, we analyze the relationship between inappropriate ads and ad SDKs. As shown in Table 1, there are 11 Google Play-certified ad SDKs that can be used in apps whose audiences include children. However, as the last column of Table 3 shows, uncertified ad SDKs, such as StartApp, AppBrain, and Facebook, are still leveraged. An interesting observation is that StartApp, a renowned ad SDK, used to be on the list of certified ad SDKs but has been removed, probably as domains and URLs sent in ads from StartApp flood users with links to malicious websites [38]. Kodular is a widely-used platform for mobile app development, and it received a lot of complaints from users [19, 23] as it continued to use the StartApp SDK for apps designed for children after StartApp’s removal. More notably, **our results indicate that even ad SDKs regarded as certified by Google Play could display inappropriate ads to children**. The most noteworthy of them is Google AdMob 7, one of the largest global ad networks.

**Case Study.** As an example, consider an app named “Berita Bola Terkini” [30], designed to share updated football news and that belongs to the Everyone age group. This app leverages Google AdMob SDK. However, our investigation discovered that Google AdMob delivered sexually suggestive and pornographic-oriented video ads in this app, e.g., the ad shown in Figure 6(a), as an attempt to lure users into downloading an adult-only live broadcast app called MoreinLive [33]. Another example is AppLovin 8, also an ad SDK certified by Google Play. An App named “HD Background Photo Editing” [32], falling in the Everyone age group and employing AppLovin SDK, was distributed with an ad shown in Figure 6(b). The ad is trying to promote a game where the polygamy experience is its major selling point (according to the Chinese slogan within the figure). This severely violates Google Play’s prohibition on ads towards children related to adult relationships. The above two cases are not accidental. We have found a total of 589 such violations in the ads displayed by Google AdMob and 48 in that shown by AppLovin.

![Figure 5: Distribution of categories of kid-unsuitable ads.](image)

---

1. https://admob.google.com/home/
2. https://www.applovin.com/
Table 4: The inappropriateness of ads in different regions based on 1,000 Android apps.

<table>
<thead>
<tr>
<th>Region</th>
<th>Category</th>
<th>US</th>
<th>Australia</th>
<th>India</th>
<th>China (Hong Kong)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td># All</td>
<td># Valid</td>
<td># Kid-unsuitable / # App</td>
<td># All</td>
<td># Valid</td>
</tr>
<tr>
<td>Ad View</td>
<td>1,027</td>
<td>9</td>
<td>4/56</td>
<td>1,144</td>
<td>1,157</td>
</tr>
<tr>
<td>Ad SDK</td>
<td>9</td>
<td>9</td>
<td>4/56</td>
<td>111</td>
<td>111</td>
</tr>
<tr>
<td>Ad-loading</td>
<td>4712</td>
<td>1,479</td>
<td>25/18</td>
<td>5,416</td>
<td>1,613</td>
</tr>
<tr>
<td>Content</td>
<td>5639</td>
<td>2,262</td>
<td>49/14</td>
<td>7,030</td>
<td>2,726</td>
</tr>
<tr>
<td>India</td>
<td>171</td>
<td>166</td>
<td>75/55</td>
<td>191</td>
<td>169</td>
</tr>
<tr>
<td>Ad-clicking</td>
<td>306</td>
<td>293</td>
<td>114/46</td>
<td>210</td>
<td>203</td>
</tr>
<tr>
<td>Content</td>
<td>33</td>
<td>28</td>
<td>42</td>
<td>7</td>
<td>9</td>
</tr>
<tr>
<td>In Total</td>
<td>11,964</td>
<td>4,456</td>
<td>266/163</td>
<td>14,962</td>
<td>5,290</td>
</tr>
</tbody>
</table>

Figure 6: Example video-form ad screenshots delivered by Google Play-certified ad SDKs.

4.4 RQ3: Region Impact on Inappropriate Ads

Our third research question is motivated by the fact that advertising violations may vary from region to region due to different cultural characteristics, child protection policies, and penalties in different countries or regions. We thus have done a preliminary study to investigate such potential differences. To build a dataset for RQ3, we select 1,000 apps from RQ1’s dataset, where the app meets the criterion of displaying at least one in-app ad in RQ1’s experiments. The selection is random while favoring apps with video ads and app promotions (the two less numerous types in the in-the-wild distribution, as shown in Table 2), as we desire to examine the regional difference in multiple ad types. We perform the experiments on the 1,000 apps four times following the same setup in RQ1, except each time under the network environment of a different region (i.e., the United States, Australia, India, and Hong Kong).

Figure 7: Distribution of inappropriate-ad categories in different regions.

Table 4 displays our preliminary investigation statistics and Figure 7 further demonstrates the distribution of inappropriate ad content detected in different regions. There are significantly more inappropriate app promotions in the US than in other regions, with HK coming second, as depicted in Figure 7. Pornographic or sexually suggestive ads exist in all regions, especially in HK, the primary forms of which are ads on explicit live broadcast platforms and sexually suggestive games. Ads containing violent content are predominantly seen in the US and HK. The advertisers there mostly seem to promote a variety of games, including PC, web, and mobile games. It is worth noting that in the US, 42 out of all 75 violation-video ads are connected to violent games portraying or simulating human-on-human violence, e.g., Mafia City and the State of Survival, far more than in other regions. Ads shown in India region have a very high number of gambling-baiting ads, such as those placed by advertisers like budsupreme.com and zupee.com. Furthermore, social media apps like Facebook, Instagram, and Now&Me, which are commonly used by teenagers and above, also frequently appear in ads in India. In addition to some violation categories shared with other regions, Australia is unique in that there are a lot of ads placed by TikTok (for 12+).

The proportions of apps with in-app ads collected in the US, Australia, and India are less than 100% (i.e., 87.6%, 88.5%, 89.8%, respectively), which could be due to network fluctuations in the ad SDK’s servers or in our test environment. Although this is only a preliminary study, the experimental results also suggest that kid-inappropriate advertising content in apps exists across multiple regions; and the distribution of different types of inappropriate content varies by region. We hope that with this preliminary study, we can shed light on this phenomenon, and inspire further research to explore the regional differences in inappropriate advertising in a more comprehensive manner.

4.5 RQ4: Comparison of AdRambler with State-of-the-art Approaches

For our last research question, we are interested in comparing the performance of AdRambler with state-of-the-art in-app ad collection efforts.

MadDroid: In 2020, Liu et al. [26] proposed an automatic framework named MadDroid for detecting malicious in-app ad content, which is the first tool that systematically collects both ad-loading content and ad-clicking content. Unfortunately, as it is incapable of accurately identifying and clicking ad views, it considers three commonly-used classes of views as potential ad views, namely WebView, ImageView, and ViewFlipper, and clicks all of them to ensure coverage. Within limited time and exploration steps, it cannot precisely click ads but only explore aimlessly, which is a significant flaw in its dynamic testing.

MAdLife: In 2019, Chen et al. [9] proposed MAdLife, another automatic approach for malicious in-app ad detection, which is also unable to identify ad views. Their solution is to open the app without any GUI exploration and only collect the WebView on
the launching page as ads, which results in a significantly lower coverage as mentioned in RQ1.

![Figure 8: Comparison with the state-of-the-art approach.](image)

We add only MadDroid to our comparison, as both our framework and MadDroid utilize app exploration for ad collection, while MAADLife adopts no app exploration and achieves a relatively lower coverage. We thus use the same configuration, i.e. number of exploring steps per app and time interval between steps, and the same dataset as MadDroid for a fair comparison. We randomly select 100 apps that are still available on Google Play from MadDroid’s dataset to avoid the apps being outdated. We run the experiment on the 100 apps with both MadDroid and AdRambler under the same network environment, and evaluate their performance based on the number of ad views collected and the number of apps with ads collected. The result is displayed in Figure 8, where AdRambler harvests 138% (54/39) ads and successfully collects ads in 121% (46/38) apps compared with MadDroid. Upon our manual investigation, we found there are mainly two reasons for MadDroid’s lower performance: (1) a number of inputs are wasted on clicking potential ad views, while most of them are non-ad views or part of ad views belonging to the same ad; this exploration flaw in particular limits MadDroid’s ability to harvest ads in apps with more than one ad; (2) as opposed to AdRambler’s depth-first search exploration strategy, MadDroid utilizes a breadth-first strategy, where the ‘BACK’ input event is given a much higher priority, preventing MadDroid from reaching ad-containing GUI pages. Interestingly, we find one case where the ad is collected by MadDroid but not AdRambler. Through further investigation, we discover that this ad is triggered by the ‘BACK’ event exclusively, which is not covered by AdRambler within a limited number of steps. Nevertheless, this evaluation clearly demonstrates AdRambler’s superiority against the state-of-the-art approach in ad collection.

5 DISCUSSIONS

5.1 Who Share the Blame?

Our empirical study demonstrates that even though Google Play-certified ad SDKs, such as Google AdMob, dominate the market share in in-app ad serving, inappropriate advertising continues to emerge in a non- incidental manner. However, it would be unfair to simply blame these certified ad SDKs alone, as the appearance of inappropriate ads could be due to the negligence and/or the unwillingness to fulfill the duties of multiple parties together: if non-certified ad SDKs are used to serve ads to children, it’s the developers’ fault for their poor choices of ad SDKs, and the app markets’ fault for their lack of education towards developers, and more importantly, their bad vetting, as they fail to identify such SDK misuse; and, if certified ad SDKs are used and inappropriate ad content is still propagated to children, the situation may be a bit more complex.

Upon our further investigations, it turns out that, to comply with app markets’ advertising policies, certified ad SDKs usually require app developers to declare the ad content rating suitable for their target audience, so that ad SDKs could serve the appropriate ads for different age groups. Thus, the inappropriate ads exposed to children through certified SDKs could be due to different reasons: if the ad developers indeed specify the appropriate ad content rating according to their target audiences, it’s the SDK’s fault for labeling the wrong ad-content rating (for instance, an ad suitable only for adults is labeled by SDKs as suitable for children), and advertisers’ fault for providing inappropriate ad creatives in the first place; otherwise, it’s the app developers’ fault for the wrong declaration. As for the app markets, they cannot perform any vetting procedures toward apps in this case, unless the declaration information is also shared with them either by developers or SDKs.

We further discovered that when integrating ad SDKs into apps, developers can specify ad content ratings with configuration settings\(^9\), either directly via the SDK’s web-side client or by invoking specific SDK APIs within their apps. We thus built a static scanner on top of Soot \(^40\) for discovering the invocations of these SDK APIs and found that none of the apps detected with inappropriate ads in our research was using these APIs. This indicates that the developers of these apps either neglected to provide appropriate ad content ratings to SDKs, or they did so on the SDK’s web-side client. Unfortunately, in terms of finding the culprits, this is the most we can do on the app side, as we don’t have access to the SDK’s web-side client, which requires logging in with the app developers’ own SDK accounts. Fortunately, through this analysis, we can provide concrete suggestions to all parties involved toward establishing effective communication channels to help find the culprits and thereby mitigate the issue in the future. We describe this in Section 5.2.

5.2 Implications

Our research in this paper indicates that children’s exposure to inappropriate in-app ads is not uncommon. When children use smartphones out of the sight of their guardians, they may be easily misled by such ads, resulting in negative cognition and even detrimental behavior. Therefore, we appeal to the community to invest more in exploring this research direction and establish a much more effective mechanism for advertising placement, distribution, and monitoring to block the dissemination of undesirable content that is harmful to children. In addition to what this paper investigates, Google Play generally prohibits ad behaviors that could lead to unintentional clicks from child users. In-app ads displayed misleadingly will result in unintentional clicks from underage users, thus undermining their experience and gaining unscrupulous profit. For future work, we aim to detect such inappropriate ad behaviors in accordance with the regulations.

Based on our analysis in Section 5.1, we make the following suggestions to all parties involved in the in-app ad ecosystem: 1) for app developers, to choose ad SDKs more carefully, preferably using only certified ad SDKs when their target audiences include children, and declare proper ad content ratings to ad SDKs and app stores; 2) for ad SDKs, to strengthen scrutiny against ad content

---

\(^9\)For example, with Google AdMob, refer to [https://developers.google.com/admob/android/targeting#ad_content_filtering](https://developers.google.com/admob/android/targeting#ad_content_filtering).
provided by advertisers, and to actively participate in the SDK certifying program provided either by Google Play or by other app markets; 3) specifically, for certified ad SDKs, to build a communication channel with the app stores and share the ad content ratings declared by app developers, so that app stores can perform checks against apps’ age groups to better maintain the appropriateness of in-app ads; 4) for advertisers, to exercise self-censorship in regards to their own ad creatives, and to directly provide accurate age-appropriate ratings themselves, as they are the best-equipped to determine suitability; 5) for app stores, to enhance their app vetting process in the following aspects: to identify uncertified ad SDK misuse utilizing static analysis approaches such as third-party library detection tools [28, 44, 45] or dynamic analysis approaches such as our AdRambler, to obtain ad content rating declared by app developers either directly from developers or from ad SDKs and compare it with the app’s intended age group to see if the app developers made the proper declaration, to implement dynamic approaches such as AdRambler to periodically detect inappropriate ads and monitor new emerging ad SDKs; and, for app stores who do not currently have an ad SDK certifying program, it is highly recommended to establish one as soon as possible.

5.3 Limitations

Although this work focuses on precisely triggering and collecting ads and detecting the presence of kid-unsuitable ads, it still comes with common limitations of dynamic testing techniques. For example, we may be unable to trigger all ads in each app within a limited scope of exploration. Also, the third-party component-provided ads shown in a certain app could be different each time launched. Nonetheless, since the content of an ad is primarily determined by the ad SDKs, ads within a single app have relatively little impact on our investigation findings, especially when we take the measure of conducting large-scale experiments to minimize the effect of the factors above. Besides, the reliability of the investigation results relies in part on the stability of the network. There are cases where network fluctuations prevent successful loading. In response, our measure is to use a VPN service[^10] with a stable Wi-Fi signal to simulate the network environment of each region. The accuracy of the Google Cloud APIs may also impact the experimental results. Fortunately, Google Cloud services are widely employed by our research community [41] and have been experimentally demonstrated in successfully flagging unsafe content [9, 26]. We also did a small-scale manual veriﬁcation with 100 random samples where the Google Cloud APIs achieved an accuracy of 75%. Moreover, as our identiﬁcation of ad views relies on a whitelist-based approach with regard to the package names of ad SDKs, it may be under inﬂuence of code obfuscation, namely the obfuscating type that renames the package names of ad SDKs. Fortunately, such obfuscation is very rare within commercial Google Play apps; we randomly sampled 100 apps from the dataset of RQ1, manually examined all the package names within the apps, and found none of those cases. Another limitation is that our framework is currently unable to explore apps automatically on the iOS mobile operating system. Fortunately, Chen et al. [10] state that the descriptions of apps for Android and iOS are identical by crawling the app metadata from the iTunes Store and Google Play, which indicates that restrictions on app age grouping apply in other formal app markets. Since ad SDKs investigated in this paper, i.e., Google AdMob, are also available for iOS app development, it is reasonable to infer that the same concern exists in iOS devices as well.

6 RELATED WORK

**Mobile Ad Content Analysis.** In 2013, a study by Chen et al. [10] first explored the content inappropriateness of in-app advertising on mobile devices from children’s perspective; since it is a manual analysis, the number of explored apps is limited to hundreds. In 2016, Rastogi et al. [34] explored the malvertising-related security risks of in-app ads. However, their study only focused on the ad-clicking content, leaving ad-loading content untouched. In 2019, Chen et al. [9] presented MAdLife, which explores abusive tactics, such as click fraud, malvertising, and other threats related to ad-clicking content. Nonetheless, its limitations are evident in ad gathering as described in Section 4.5. In 2020, Liu et al. [26] developed MadDroid to detect malicious ad content as introduced in Section 4.5, which is unable to precisely identify and click the ad views in dynamic testing. While these existing efforts are more or less relevant to ad collection, none of them achieves ad view identification during automated app exploration, or focused on automatically detecting ad inappropriateness towards children.

**Child-related Mobile App Analysis.** Liu et al. [25] designed a machine learning model for predicting whether mobile apps are designed for children. Reyes et al. [35] proposed a dynamic analysis framework to evaluate privacy-related behaviors of Android apps, which is then used to check the app’s compliance with the Children’s Online Privacy Protection Act (COPPA). Luo et al. [27] proposed an automatic approach to determine whether a kid-friendly Android app contains inappropriate content, including a very rough inspection of ads based on screenshots only. However, none of them have specifically concentrated on the inappropriateness of in-app ad content toward children.

7 CONCLUSION

In this paper, we examine the issue of children’s exposure to inappropriate advertising in apps whose target audiences include children. To support our large-scale investigation, we first characterize the regulations that apps with children audiences must comply with, and then propose a dynamic analysis framework named AdRambler to automatically analyze the advertisements in mobile apps, gathering both the ad-loading content and the ad-clicking content, and further detecting inappropriate content based on the collected ads. With the help of AdRambler, we identify 1,289 ad violations from 775 apps and reveal that even ad SDKs certified by Google Play may distribute ads that are not suitable for children. We empirically find out the in-app advertisements are non-occasionally not in compliance with the apps’ age groups. Finally, we discuss the potential culprits of this issue and make concrete suggestions to all relevant parties to take more proactive measures in safeguarding children from inappropriate advertising.

ACKNOWLEDGMENT

This work is partly supported by ARC Laureate Fellowship FL190100035 and the Guangdong Basic and Applied Basic Research Fund (Grant No. 2021A1515011562).\[^{10}\]https://nexitally.net/
REFERENCES


[34] Vaibhav Rastogi, Rui Shao, Yan Chen, Xiang Pan, Shihoung Zou, and Ryan Riley. Are these ads safe? Detecting hidden attacks through the mobile app-web interfaces. In NDSS, 2016.


