Understanding GDPR compliance of tracking pixel declarations using privacy filter lists

Bachelor Thesis

Author(s):
Ganz, Rita

Publication date:
2022-02-18

Permanent link:
https://doi.org/10.3929/ethz-b-000535362

ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Understanding GDPR compliance of tracking pixel declarations using privacy filter lists

Bachelor Thesis

Rita Ganz

February 18, 2022

Advisors: Karel Kubíček, Prof. Dr. David Basin
Department of Computer Science, ETH Zürich

Abstract

Tracking and data collection on the World Wide Web are ever-present. Privacy laws, such as the *General Data Protection Regulation (GDPR)* of the European Union, have come into effect to address user tracking. To comply with GDPR, websites use consent banners informing users about tracking and giving acceptance choices; these are server-side tracking-prevention mechanisms. Independently, ad- and tracking-blockers provide client-side protection. We investigated tracking pixels and their treatment by consent declarations and filter lists to enhance our understanding of privacy risk and GDPR compliance.

We collected tracking pixels and images together with their purpose labels from websites with detailed consent notices. Using this dataset we trained an XGBoost model to predict purposes with mean balanced accuracy of $95.84 \pm 4.20\%$. We compared classifications by the model, the consent declarations, and the filter lists, finding strong overall agreement, but also wrongly blocked normal images and rare trackers missed by filter lists. Our model and data collected from consent declarations may help improve privacy lists.

In our dataset, 62.8% of sites contained possible tracking pixels that were not found in consent notices, 5.9% of sites contained pixels matched to a potentially wrong outlier label, and 0.7% of sites declared a wrong purpose for Google Analytics pixel, all of which may indicate possible GDPR violations.

Acknowledgment

Thanks first and foremost to Karel for giving me the chance to work on this fascinating topic and all the invaluable support and help along the way. Thanks to Prof. Basin for welcoming me in his research group. Many thanks to Guy for proofreading the final draft of this thesis.

Contents

Chapter 1

Introduction

Most people would probably not want the world to know all their interests, political views or even illnesses. They might not even realise how much of this information they involuntarily share when browsing the web. Tracking and data collection on the World Wide Web are ever-present, e.g., [8]. It is often not transparent to the user what data is collected by whom, and what it is used for. It is a privacy concern in itself that this data might contain sensitive information, including identity, geographical location, faith, or political orientation. Targeted actions based on this data could, and have led, to manipulation of people and potentially 'democratic crisis' [15].

To protect users privacy different actors followed different courses. Governments brought in privacy laws, such as the *ePrivacy Directive* [10] and later the *General Data Protection Regulation (GDPR)* of the European Union (EU) [11]. Individuals and groups of people who were concerned about their privacy, or who became annoyed by advertisements on the World Wide Web, created tools to block tracking and adverts in the browser, such as AdBlock/uBlock using EasyPrivacy and EasyList [12]. Both, in different ways, assess the privacy risk of a certain tracking activity.

GDPR affects any organisation world wide that targets people in the EU or collects their data [11, Article 3]. It states that the host of a website or a third party needs a legal basis for data collection. The most commonly used legal basis is consent of the user. Consent has to be freely given, specific, informed and unambiguous [11, Article 4(11) and 7]. As a reaction to GDPR, websites often use so called *consent notices* or a *consent banner* that gather consent from the user and thus provide a legal basis for data collection. It is legally demanding for a website operator to ensure compliance of the consent notice, many therefore rely on a third-party solution, a so-called *Consent Management Platform (CMP)*. The CMP handles the legal terms and conditions for privacy policies, implements the consent banner, collects consent from the user, and redistributes it.

[11, Recital 32] states that if the collection and processing of user data has multiple purposes, consent should be given for all of them. This purpose and consequently the data collection can be more or less privacy threatening. The purpose labels assigned to different trackers in consent notices can therefore serve as a proxy to assess potential privacy risk.

Filter lists are a set of rules to recognise, for example, adverts or trackers on the internet. They are often maintained by human operators [27]. In the case of EasyPrivacy, whose objective it is to remove tracking [12], the decision which content to block is also an assessment of privacy risk, however, a binary one. Comparing this to the purpose labels can help to understand GDPR compliance and might ultimately enable the detection and prosecution of violations on the one hand, and the improvement of said privacy filter lists, on the other.

1.1 Our Study

One way to track users is through tracking pixels. Tracking pixels are small images, that can be incorporated in a website in different ways, i.e., as an image in an <img>-tag, inside an IFrame or JavaScript [28]. When the website is loaded, the tracking pixel is requested from typically a third-party server it is on, like any image. In contrast to a normal image, a tracking pixel will send information that can be used to identify and track the user to the server. In this study we focus on tracking pixels and their treatment by consent declarations and filter lists to understand their privacy risk and GDPR compliance.

To achieve this, we need to assemble a data set containing tracking pixels found in the wild and their corresponding category labels. Previous work on cookies [4] has illustrated that a promising approach is to crawl a list of websites and collect the labels from consent notices provided by CMPs as well as the cookies found on the web page. We adapted this approach for tracking pixels and describe it in detail in chapter 3. Once the data set is collected we identified features of the pixels, extracted them, and trained a classifier using XGBoost, see chapter 4. This classifier is used together with information collected from privacy filter lists to analyse different aspects of tracking pixels and potential violations of GDPR compliance in chapter 5. As the first part of this study builds heavily on previous work by Bollinger et al. [2, 4, 3], we summarise previous work and design choices to then discuss the additional challenges of incorporating tracking pixels into the framework in each appropriate section.

We collected 813’162 pixels or images from 10’380 domains that contained the CMP Cookiebot. Of these samples we matched 129’824 observed pixels to a consent declaration. The remaining 683’338 samples are either normal

images, undeclared pixels, or pixels with non-standard declarations that failed to be matched.

All samples were then used to train an XGBoost classifier to distinguish between the four purpose categories. The model achieved a high average and mean balanced accuracy of 98.77 $\pm$ 0.03% and 95.84 $\pm$ 4.20%, respectively. The resulting model has three applications: identification of important features of tracking pixels, assessment and maintenance of filters, and detection of potential violations.

Our inspection of the model helped to detect important features to recognise tracking pixels by. Specific important features from the URL of tracking pixels are common domains names, such as `google-analytics.com`, keywords used in the path such as `collect`, or presence of specific query parameters. Generic important features are size, particularly whether the image is 1x1 pixel large, and a high amount of entropy in the URL or headers. This is often caused by pseudo-random strings that could serve as unique identifiers to track a user.

We compared the assessment of privacy threat through the filter lists EasyList and EasyPrivacy, and through the purpose declared in consent declarations and by our model. We found strong agreement overall, but also identified cases of wrongly blocked normal images and trackers missed by filter lists. Our approach was able to detect rare tracking pixels that were missed by the filter lists (false negatives of the filters). Thus, using our data collection from consent notices might help privacy list maintainers to find rare pixels and keep their lists up to date. We also found wrongly blocked images by too broad rules in the filter (false positives).

We show that tracking pixels can be larger than 1x1 pixel, such as the 2x2 pixel used by Adobe Analytics. This illustrates that limiting a study to only pixels of size 1x1 might systematically miss certain pixels.

Taking into account both the decisions made by filter lists, a majority vote on purpose declared by the consent notices, and our model's prediction can help to confidently detect tracking pixels. It also could be used to assign categories to samples that could not be matched to a purpose declaration, or to detect potential violations of GDPR compliance.

Revisiting our dataset from the point of view of potential violations to GDPR, we discuss ways of detecting violations. We find that between 1.5 and 12% of samples that are either considered strictly necessary pixels, potentially undeclared pixels, or normal images, could be tracking pixels used for analytics or advertising purposes; and therefore might be possible violations. We found pixels matched to a consent declaration with a purpose label deviating from majority class, and therefore possibly wrongly labeled, on 618 different webpages (5.9% of pages visited). More specifically, we found

wrong "necessary" purpose labels for the well known Google Analytics pixel on 74 websites (0.7% of websites). Of the 10'380 domains we analysed, 6519 or 62.8% contained potential tracking pixels (i.e., images of size 1x1) that could not be matched to a consent declaration and might not be declared.

1.2 Our Contributions

ML model: We develop an ML model that classifies images into purpose categories. We use this model to analyze how images track users and to assess filters and pixel declarations.

Filter list assessment: We inspect false positives and false negatives of filter lists.

Detection of violations: We describe three methods for detecting potential violations present in the consent banners. We identify that between 0.7 and 62.8% of websites contain such potential violations.

Outlier tracking images: We identify tracking images that are larger that 1x1 pixel. This shows that focusing on 1x1 pixels as in past studies is insufficient.

Chapter 2

Background

2.1 Tracking Pixels

Tracking pixels are commonly defined as small images of size 1x1 pixel, e.g., [13, 24], and are also known under the names of 'invisible image', 'invisible pixel', 'pixel tag', 'web beacon', or 'web bug' [24]. Tracking pixels make use of the way images are incorporated and loaded in webpages. A user's client visits a webpage and loads all the images including the tracking pixels. To do so, it requests them from a server where the images are stored. Some information is often passed on to the image server, for example, the screen size, to determine in which resolution the image should be returned. The difference between loading a normal, i.e., non-tracking, image and a tracking pixel, is that when requesting the tracking pixel, information that can be used to identify and track the user is sent to the server.

In theory, one could use any image to track users, but small images have the advantage of incurring little loading delay and of being almost invisible for the user. To enhance the invisibility they could also be transparent or the same colour as the background of the website. As part of this study we aim to verify if tracking pixels truly are of size 1x1 pixel as is commonly assumed, e.g. see [13, 24]. We will discuss the results of this in 5.3.1.

Not every image of size 1x1 pixel is a tracking pixel. Small and invisible images could for example also be used to position elements on a webpage[1].

2.2 GDPR, Consent notices, and CMPs

The General Data Protection Regulation [11] (*GDPR*) was adopted in 2016 and put into effect in 2018 as EU's data protection law. GDPR affects any

[1] https://www.yourhtmlsource.com/images/singlepixelimages.html accessed: 24.01.2022

organization anywhere that targets people in the EU or collects their data [11, Article 3]. We will here only discuss some aspects of the GDPR relevant for this study.

The host of a website or a third party needs a legal basis for collecting data about a user that is not strictly necessary for the service requested. As such counts consent of the visitor, if the data is necessary for a contract including the user, legal obligation, protection of vital interests of the user, public interest or legitimate interest [11, Article 6a-f]). Consent has to be freely given, specific, informed and unambiguous [11, Article 4(11) and 7]. This implies that pre-ticked boxes or inactivity does not mean consent [11, Recital 32]. The purpose of the data collection has to be assigned to usage categories, so that consent can be given specifically per category. [11, Recital 32] states that if the collection and processing of user data has multiple purposes, consent should be given for all of them. This also implies that the user must have the option to deny consent for certain purposes.

Consent notices or consent banners do exactly as their name suggests, they gather consent from the user and therefore provide a legal basis for the collection of data. They have existed in various forms before GDPR. However, to make it easier for website hosts and actors in the tracking and advertising industry to meet the requirements on consent, (freely given, specific, informed and unambiguous) the Interactive Advertising Bureau Europe (*IAB Europe*) has designed the Transparency and Consent Framework (*TCF*) . The TCF contains implementation guidelines and introduced the notion of a Consent Management Platform (*CMP*). In exchange for a service fee the CMP handles the legal terms and conditions for privacy policies, and implements the consent banner displaying third-parties involved and the purpose of the tracking technologies used. The CMP collects consent from the user and redistributes it to advertisers and trackers including the third-parties.

2.3 Filter Lists

Filter lists are a set of rules to recognise for example adverts or trackers on the internet. They can be used in ad-blocking browser extensions to block specific web requests when loading a webpage. Each filter rule is structured like a regular expression. Additional options allow to adjust the scope of the filter, e.g., to only apply the filter if the domain is third party or if the request is for an image [27, 23]. Filter lists can be categorized into community-driven, centralized or algorithmic. The difference between community-driven and centralized is mainly in who manages and curates the list, in both cases humans add rules to the list. A filter list can also be derived algorithmically, using heuristics to detect trackers automatically [27].

The EasyList filter lists were originally designed for Adblock [12]. The

EasyList authors maintain different lists serving slightly different purposes. While the EasyList filters must remove adverts, and may incidentally remove tracking, the EasyPrivacy filters must remove tracking [12]. EasyList is community-driven [27] and rules are updated, added or removed based on suggestions from users.

Chapter 3

Tracking Pixel Crawler

As mentioned before, a promising way to collect a data set containing pixels and purpose labels is to crawl a list of websites that employ a CMP, extract the purpose labels from the consent declaration and extract the pixels from the website including subpages. As Bollinger [4] discussed for cookies this allows the collection of labels for unique cookies from the host that created them. Such cookies might, for example, not be contained in a database such as the *Cookiepedia* repository [22], which is a publicly accessible database containing cookies and their purpose label. A disadvantage to collecting purpose labels from CMP consent notices is that they may be wrong or inconsistent as the labeling is done by a large set of human operators with varying levels of competency, or possibly even motivation for intentional mislabeling [2, Sec. 7.1]. From an outside perspective it is not easy to infer the true purpose of a cookie, due to limited information, however, one can estimate the noise in the data by assessing whether the same cookies have been labeled differently by different operators [2]. The same considerations are of course true for collecting tracking pixels and their purpose labels from CMP consent notices. In addition, there is no publicly accessible equivalent repository to *Cookiepedia* for pixels, and their purpose labels, that we could use. In section 4.3.3 we will discuss further the effect of noise in our data set.

The web crawler we used is based on previous work by Dino Bollinger [2], which was subsequently minimally modified to include tracking pixels. The web crawler consists of two main parts. First a fast crawl that checks presence of the CMP Cookiebot, OneTrust and Termly on the chosen domains. The CMP presence crawler returns a list of websites for each of the CMPs and additional lists of failed URLs or websites that timed out during the crawl. In a second web crawl only websites containing the CMP Cookiebot were visited and the consent declarations as well as the pixels were collected. In the following sections we first discuss what we target with our crawl, followed by details on the two web crawlers and their results.

3.1 Crawler Targets

3.1.1 Chosen Domains

Website popularity rankings are a common resource for researchers conducting studies on privacy practices in the wild. We use Tranco [19] as it combines ranking data from different sources, i.e., Alexa, Cisco Umbrella and Majestic, and provides a higher stability and resilience against manipulation than any of the three sources on their own. In addition, Tranco provides a permanent record of the lists, which is particularly important for research as it allows other researchers to replicate a study [19].

We use two Tranco lists [19], one for European domains[1] and one for worldwide domains[2]. We only include pay-level domains, but multiple domains for a certain organisation that only differ in top level domain, e.g., site.fr and site.de would both be included. We used the top 1 million domains from the worldwide list plus 465'867 domains hosted in Europe and not contained in the top 1 million worldwide domains. The reason for including these European domains is that we expect them to be more likely to contain CMPs as they have to comply with GDPR.

3.1.2 Chosen CMPs

There are a great number of CMPs offering their own website plugins resulting in a great variety of cookie banners, from simple notifications to banners that let the user choose from dozens of possible categories. As we want to match tracking pixels observed in the wild with declared purposes, we need to be able to extract pixel specific purpose declarations. Therefore a suitable CMP has to fulfil the following two conditions as discussed by [4]: first, the purpose labels must be accessible in a computer-readable format, ideally hosted remotely on a server by the CMP itself; second, the CMP has to list purpose for each cookie or other tracking technology publicly and reliably on every website where the plugin is correctly implemented. This means it needs to include some sort of identifier, e.g., name, for the tracker in the consent declaration. In this study we additionally require that the CMP declares tracking pixels as such. It is worth noting that many CMPs, which declare purpose for each tracker, do not specify the kind of tracker. In these cases, our outsider perspective does not easily allow us to infer whether a declaration is for a pixel.

We built our assessment of suitable CMPs based on table 1 from [4] and added our third criterion. The results are listed in 3.1. The market share of the CMP, whether or not the consent declaration can be remotely accessed

[1] Tranco Europe dated 04.12.2021, available at https://tranco-list.eu/list/Q574
[2] Tranco list dated 03.12.2021, available at https://tranco-list.eu/list/PJWJ

(the first condition) and whether or not the CMP provides labels for each tracker (second condition) are taken from [4]. They are therefore as of October 2020 the CMPs with the highest market share worldwide as reported by the technology trend database BuiltWidth [5].

In this study we only consider CMP Cookiebot. Cookiebot has a high market share and fulfills all three conditions. One other CMP we could have selected is Termly. However, as expected from its lower market share, we found Termly present only on very few domains, i.e., on 531 domains, in our presence crawl, see table 3.2. Dino Bollinger [2, Sec. 6.1.3] also found that Termly declarations are very rare, and in addition, more than half of all cookie declarations he found did not assign a purpose, but were "unclassified". For Cookiebot we found that pixels only make up a small proportion of trackers declared, see table 3.4, which would likely also be the case for Termly. We excluded Termly because it is unlikely to provide a large enough return in training samples to justify the time investment.

3.1.3 Chosen Purpose Categories

We use the same purpose categories for the tracking pixels as [4] used for cookies. These correspond to the categories defined by the UK's International Chamber of Commerce [21]:

1. Strictly Necessary
2. Functional
3. Analytics
4. Advertising

As Dino Bollinger outlined in detail in [2, sec. 3.2] these categories should represent a user's consent preferences well and also be intuitively understandable to an average user of the browser extension. Potential privacy threat increases in the order listed. He also chose these purpose categories for practical reasons as they closely resemble the categories used by the CMPs of interest.

Our starting point was that these categories also make sense for pixels. The ePrivacy Directive [10] recognises "strictly necessary" as a purpose category for pixels. Cookiebot and Termly, the two CMPs we found to explicitly list pixels in their consent notices, both use these four categories to classify them; hence these companies must see sense in it. However, given that we did not find many truly necessary or functional pixels we will further discuss whether the categories truly make sense for pixels in section 5.2.

The purpose of "strictly necessary" pixels is to provide a core functionality for a website. "functional" pixels are not essential but can be used to customize a

Table 3.1: CMP market share as reported by BuiltWith [5] and suitability based on our three criteria: consent declaration is remotely accessible, contains labels for individual trackers and declares pixels as such.

CMP	Market share	Remote?	Labels?	Pixels?
Cookie Consent by Osano	2.25%	✓	✓	✗
Cookie Notice	1.29%	✗	✗	✗
OneTrust	1.17%	✓	✓	✗
OptAnon	1.08%	✓	✓	✗
Cookie Law Info	0.95%	✗	✗	✗
Cookiebot	0.77%	✓	✓	✓
Quantcast CMP	0.68%	✓	✗	✗
UK Cookie Consent*	0.33%	✗	✗	✗
TrustArc Cookie Consent	0.26%	✓	✗	✗
WP GDPR Compliance	0.20%	✗	✗	✗
Moove GDPR Compliance	0.18%	✗	✗	✗
tarteaucitron.js	0.16%	✗	✗	✗
Usercentrics	0.16%	✓	✗	✗
CookiePro	0.15%	✓	✓	✗
Borlabs Cookie	0.12%	✗	✓	✗
Eu Cookie Law	0.12%	✗	✓	✗
Cookie Script	0.07%	✓	✓	✗
Cookie Information	0.06%	✓	✓	✗
Termly	0.05%	✓	✓	✓
Cookie Info Script	0.05%	✓	✗	✗
Easy GDPR	0.04%	✓	✗	✗

* now operated by Termly

website. The purpose of "analytics" pixels is to analyse website performance and they should only track a user on a single domain. They can, however, be used to track a user across multiple devices. A very common example of this are Google Analytics pixels. If, for example, the user is logged in to her google account on both her mobile phone and her computer, and visits the same domain on both devices, these visits can be linked. The purpose of "advertising" pixels is to deliver an advertisement product. They can be used to track the user across multiple websites, such as the Facebook pixel.

The CMP of interest, Cookiebot, also has the category "unclassified/unknown" for pixels or other tracking technologies. For cookies for which a purpose cannot be determined automatically Cookiebot states that the host can either request the purpose from the third party or classify the cookie

manually[3]. Assuming this is similar for pixels, "unclassified" should be a temporary category until the purpose is known. The user cannot reject this category. It is therefore of interest when assessing compliance with GDPR, particularly as Bollinger [2, sec.7.4] found that unclassified cookies were observed on some websites, as if they were consented to, but not on others. This category is not used for the classifier.

3.2 CMP Presence Crawler

3.2.1 Design and Purpose

Ultimately we aim to collect purpose labels of pixels for domains hosting Cookiebot. On BuiltWith's ranking of CMPs Cookiebot appears in less than 1% of the domains contained in the top 1 million sites [5]. This indicates that we can only find the data we are interested in on a small subset of the chosen domains. In a first step we therefore asses reachability of all the chosen domains and whether the landing page contains Cookiebot. This is done using Python `request` library, which is both fast and can be highly parallelised [2, sec. 3.3.1]. It should be noted that this approach will not detect presence of a CMP if first a JavaScript needs to be executed and might wrongly detect a CMP if for example a consent notice was insufficiently removed; however, both are expected to be rare [2, sec. 3.3.1]. We used the CMP presence crawler from Dino Bollinger [2] as it provides all the functionality we need. Previous studies, e.g., Eijk et al. [26], have shown that websites may display different consent notices or not display them based on the geolocation of the user. To see the same consent notices as a user from the EU, we conducted the crawl through a VPN located in the Netherlands.

3.2.2 Results

The CMP presence crawler was run on a total of 1'465'867 domains and took 47 hours on a consumer-grade laptop. Table 3.2 shows the results of the CMP presence crawl. For this study we are only interested in domains with CMP Cookiebot. The stated numbers for Cookiebot are after duplicate domains were removed from the list, which is the reason why the total is not the sum of domains found from the European and worldwide list.

In the next step we will use the 10'380 domains, which likely contain Cookiebot CMP, to collect consent declarations and pixels.

[3] https://web.archive.org/web/20201111204915/https://support.cookiebot.com/hc/en-us/articles/360003735214-Unclassified-cookies-how-do-I-classify-them-manually-, accessed on 09.02.2022

Table 3.2: Results of the CMP presence crawl of the target domains.

Domain List	Failed URLs	No CMP	Cookiebot	Termly
Tranco Worldwide	244'871	691'994	6101	422
Tranco Europe	71'919	378'815	4318	109
Total	316'790	1'070'809	10'380	531

3.3 Consent Crawler

3.3.1 Design and Purpose

As mentioned above, the consent crawler is designed to collect the actual data set later used to train a classifier by crawling the pre-selected domains returned by the presence crawler. The consent crawler was implemented by Dino Bollinger [2] and is built on top of OpenWPM v0.12.0 [7]. OpenWPM is a web privacy measurement framework and has been used by over 75 studies into web privacy as of 23.12.2021. It uses Firefox and automation provided by Selenium. OpenWPM provides the functionality to collect different types of data, for example cookies or HTTP-requests, redirects and responses. It stores this data in a SQLite database. Dino Bollinger added functionality to collect information from the consent notice and also installed the Consent-O-Matic extension [18] into the browser profile. The Consent-O-Matic extension interacts with the consent notice and ensures that all categories of cookies and other trackers are accepted and can be collected by the crawler [2, sec. 3.3.2]. For the same reason, the browser settings were chosen such that the browser does not block any trackers.

The information collected from the consent notice is stored in a SQL-table in the same database as the other information OpenWPM collects. The information collected from the consent notice contains the fields listed in table 3.3 [2]. Except for expiry, which is cookie specific, all fields are also important for tracking pixels, as they let us filter the table for relevant information (e.g., we are only interested in type_name "Pixel") or join it to other tables. Database processing will be discussed in section 3.4.

3.3.2 Adding Tracking Pixels to the Consent Crawler

There are at least three different ways to include tracking pixels, and images more generally, in a website: as an image in an <img>-tag, inside an IFrame or JavaScript [28]. It is straightforward to collect images and pixels if they are included in the HTML document via the <img>-tag, for example with jQuery or the Selenium webdriver, which can be accessed inside the OpenWPM framework. An advantage of this approach is that we could also collect attributes such as the dimensions of the image, whether an image is transparent, or alternate text, which could be useful features. However,

Table 3.3: Information collected from the consent notice.

Field	Description
id	uniquely identifies each entry in the consent data table
browser_id	identifies the browser instance, which collected the data
visit_id	uniquely identifies the website that was visited
name	name of the tracker as specified by the consent notice declaration
domain	origin of tracker as specified in the consent notice declaration
cat_id	internal category index
cat_name	name used in the consent notice declaration to describe the purpose
purpose	short text describing the purpose of the tracker or how it works
expiry	declared expiration time
type_name	name of the type of tracker as specified by Cookiebot
type_id	index of the tracking technology type.

this approach misses the majority of tracking pixels, as also observed by Fouad et al. [13]. It is also noteworthy that some providers of tracking pixels mention that IFrames are the best option for including pixels [28]. Manually checking the developer tools for a few websites that use tracking pixels revealed that, Facebook and Google Analytics pixels, for example, are loaded by scripts. It is not possible to access the content of third-party IFrames or scripts with jQuery or JavaScript. We considered monitoring changes to the Document Object Model, DOM, which is a common solution to detect dynamically loaded images. However, a request for an image can be sent without changing the DOM, making this an unreliable means of detecting pixels.

Taken together this implies that the best option to collect all the tracking pixels and images is to monitor HTTP traffic and filter for images requested and received, even though this limits the features we are able to collect. OpenWPM contains the instruments to collect all HTTP requests, redirects and responses and stores them in a SQLite database. It can further collect content returned in HTTP responses, such as images, in a LevelDB database. A LevelDB database stores key-value pairs as byte arrays [14]. OpenWPM calculates the MD5-hash of the content, which is stored in the table http_responses, and serves as the key for the LevelDB database to retrieve the content data. We only collected images to save storage space, as other content was not required for this project. As we also needed to collect consent banner data we used the web crawler from Dino Bollinger's work [2], and activated the http_instrument and save_content instrument for images.

The consent crawl therefore works as follows: Given a list of URLs to crawl, a URL is sent on a first-come, first-serve manner to a browser instance. The browser instance connects to the website's landing page and tries to extract the data from the consent notice. If it is successful, it collects the images from the landing page by monitoring HTTP traffic. It then accesses five subpages and also collects the images there.

The consent crawler was again run through a VPN located in the Netherlands to ensure seeing the same consent notice a user from the EU would.

3.3.3 Results

The presence crawl used seven parallel browser instances and took a little over 78 hours to complete. We crawled all the unique 10'380 Cookiebot domains found by the CMP presence crawl. Consent declarations were found on 8722 of the 10'380 domains (i.e., on 84.0% of domains crawled). This is comparable to [2, sec. 6.1.2] which reported successful data extraction from 8272 of 9782 domains (84.5%) for Cookiebot. We collected a total of 449'955 consent labels of which 39'387 (8.8%) were pixels. Table 3.4 gives an overview of how many pixels and other trackers, such as cookies, were found per category.

In absolute counts we find fewer declarations for all trackers than [2, sec. 6.1.2], which is not surprising, given he collected data for three CMPs including Cookiebot and we only report results for Cookiebot. Comparing table 3.4 to [2, sec. 6.1.2] we find a similar pattern in the distribution of labels for all trackers as Dino Bollinger reports for cookies. Disregarding the category uncategorized/ unknown, both studies find "advertising" to be the most common category, followed by "strictly necessary" and "analytics", which make up a similar fraction, and lastly "functional". There are differences however: we find a lower percentage of "advertising" (48.6% vs. 55.94%), a higher percentage of "strictly necessary" (17.9% vs. 11.82%) and "analytics" (16.7% vs. 12.64%) and a lower percentage of "functional" (3.5% vs. 9.90%) than Dino Bollinger. We further report a higher percentage of unclassified/ unknown trackers than Bollinger did for cookies (13.2% vs. 9.7%).

Comparing the distribution of labels across categories for pixels with those found by Bollinger for cookies [2, sec. 6.1.2] it is most noticeable that "functional" and "strictly necessary" pixels are far rarer than "functional" and "strictly necessary" cookies, respectively.

We collected 1'063'638 HTTP requests for images including pixels. Counted is only one HTTP request per resource, i.e., the first request for a specific image. Subsequent requests due to redirects are not counted. We collected 900'930 HTTP responses containing images.

Table 3.4: Number and percentage of consent labels found per category for all kinds of trackers declared and for pixels only.

Label	Pixels	All Trackers
Strictly Necessary	1012 (2.6%)	80'612 (17.9%)
Functional	57 (0.1%)	15'895 (3.5%)
Analytics	6362 (16.2%)	75'241 (16.7%)
Advertising	29'200 (74.1%)	218'622 (48.6%)
Unknown/Uncategorized	2756 (7.0%)	59'585 (13.2%)

3.4 Database Processing

As discussed above, the consent crawl provides us with all the information needed for the classifier, but not in a format we can use directly. To assemble the training data set we first filter the data.

1. Consent notice declaration: we are only interested in consent declarations for pixels.
2. HTTP requests: we are only interested in requests for images and we are only interested in the first HTTP request for an image. It is important that we are able to block the first request for a tracking pixel in the browser extension, as this request already provides tracking information about the user.

The much larger challenge was to then match records from different SQLite tables and combine them with the data stored in the LevelDB. This will be discussed next.

3.4.1 Matching Declared Pixels to Observed Pixels

In a first step we need to match declared pixels from the consent declaration with observed pixels from the HTTP requests. Unlike cookies, pixels do not have an identifier that would allow us to easily link a consent declaration to an observed pixel. In the consent notice banner each pixel is given a name. We could not find conclusive information on how that name is determined. It seems that Cookiebot stores the names of all Cookies and pixels in a central repository and only these names can be used in the consent banner. Customers of Cookiebot can, however, ask for a name to be added to that repository.[4] However, this does not seem to prevent some declarations from containing unlikely names such as `pagead/1p-user-list/Use%20Google%20Tag%20Manager%20to%20install%-20the%20Google%%20tag.%20Google%20Tag%20Manager%20makes%20it%20e-`

[4] https://support.cookiebot.com/hc/en-us/community/posts/360017641714-Cookies-with-dynamic-names, accessed 23.12.2021

asy%20to%20update%2C%20add%2C%20and%20manage%20your%20website%20-tags%20without%20editing%20your%20code.%20%20Once%20a%20c.

We could not find any naming conventions for pixels while Cookies seem to have a clearly defined name a website operator could look up.[5] We observed that often the name of a pixel is part of the path or domain of the URL of a corresponding pixel. In lieu of a better option we used this to match a consent declaration to an observed pixel, i.e., an HTTP request for an image, together with checking if the netloc of the URL matched the domain given in the consent declaration.

As a second step we need to link the consent notice and observed pixel from HTTP requests to actual pixels returned in the HTTP response and stored in the LevelDB. We can link an HTTP request and the corresponding response using that they have the same visit_id and request_id[6]. From the HTTP response table we can then extract the key to look up the image in the LevelDB. We further extract information from the image itself, such as format, colour mode, size in pixels and colour including transparency.

We additionally included all images without matching consent declaration in the "necessary" category. Images of course do not require a consent declaration, but they provide a core functionality for the website. There is no a priori way to tell if an HTTP request for an image-resource is for a tracking pixel or a normal image. If a classifier was later to be used in a browser extension to block HTTP requests for certain categories of tracking pixels without breaking a websites core functionality, it has to be able to recognise normal images as "necessary".

Including everything unmatched in the "necessary" category has the potential pitfall of adding undeclared tracking pixels or tracking pixels that are declared but failed to be matched to a consent declaration. As a first indication of whether this is the case we counted the number of potential pixels, i.e., images of size 1x1. We observed that a large number of them came from just nine netlocs that are from well known trackers (see appendix A.2 for details). We moved those to their appropriate category to mitigate the rather coarse approach of putting everything unmatched into "necessary". In section 5.2.1 we will discuss whether there are true "functional" pixels and find that most pixels declared as "functional" should be in another category. We therefore also moved the same well known trackers out of the "functional" category.

For both models the matched training data set is stored as JSON format using samples from the "necessary", "functional", "analytics" and "advertising"

[5] https://support.cookiebot.com/hc/en-us/articles/360018481879-Manually-adding-cookies accessed: 23.12.2021

[6] Even though OpenWPMs documentation specifies that request_id changes in case of a redirect, this was never observed in our dataset.

category only. Each observed pixel is an object with attributes inside the JSON. The JSON is then used for feature extraction in the classifier and to gain further insights into the dataset.

3.4.2 Results

We extracted 129'824 training samples from matching consent declarations to HTTP requests and responses, and extracting the image information from the LevelDB. These samples are a subset of the samples matched to consent declarations, due to errors occurring when accessing the LevelDB (217 errors) or due to a missing HTTP response for a request. Of all these samples 72 (0.06%) were larger than one pixel.

We then added 683'338 training samples to the "necessary" category that could not be matched to a consent declaration. Of these 107'431 (15.7%) are of size 1x1 and might possibly be tracking pixels. Based on their origin (see A.2 for details) we moved a total of 67'719 samples out of the "necessary" category: 23'251 to "analytics", and 44'468 to "advertising", respectively. For the same reason, we also moved 59 samples out of the "functional" category, 2 to "analytics", and 57 to "advertising". All these moved samples are potential examples for violations of GDPR. We will elaborate on this in section 5.4.

In total we therefore extracted 813'162 training samples. Table 3.5 gives an overview of the final number of samples per category.

Table 3.5: Number of training samples per category.

Label	Number of Samples	Percentage of Samples
Strictly Necessary	619'569	76.2%
Functional	49	<0.1%
Analytics	63'043	7.8%
Advertising	130'501	16.0%

Chapter 4

Tracking Pixel Classifier

To use the data collected by the webcrawler to train a classifier, we must transform it into a numerical embedding. First we discuss what information we have collected about the pixels in more detail. Then we discuss the features we can extract from this and how to do it. Next we use this numerical embedding to train the classifier. We end the chapter with an evaluation of the classifier.

4.1 Feature Engineering

The goal of the feature engineering is to have a uniformly-sized real-valued vector representation for all pixels as this can be used as input to a range of classifiers. In contrast to cookies [2, sec. 4.2], we only have "per-pixel features" as there are no updates to a pixel. There may be multiple requests for the same pixel due to redirects. We do, however, only consider the first request for any image resource as the tracking can already happen here. Thus, if one was to block tracking by pixels through a browser extension, one would need to block the first request. For each image or pixel we have collected information on the following.

1. The *URL* of the image resource requested contains the netloc, i.e., the domain and subdomains if present, and the path to the image resource. It may contain different query parameters and their values. For example, the URL `https://www.example.com/sites/image_1.PNG?itok=PBvXT` would have netloc `www.example.com`, path `/sites/image_1.PNG` and contains one query parameter called `itok` with value `PBvXT`.
2. The *headers* of the HTTP request can contain different header fields, which may or may not be present, and their values. We found that all HTTP requests in our training samples contained the header fields 'Host', 'User-Agent', 'Accept', 'Accept-Encoding', 'Accept-Language'

and 'Connection'. A further four header fields were only present in a fraction of the 813'162 training samples. 'Referer' was found in 679'098 (83.5%) training samples, 'Cookie' in 390'851 (48.1%), 'Origin' in 468 (0.06%), and 'Alt-Used' in 25 (0.003%) samples.

3. The actual *image* returned in the HTTP response. From the image we can extract the following:
   a) The *size* given as width and height of the image in number of pixels.
   b) The *colour* in RGBA colour values of the tracking pixels or for larger images of the first pixel. For images which did not use RGBA colour mode the image was first converted to RGBA.
   c) The *format* of the image. Observed image formats were: GIF, JPEG, TIFF, CUR, MPO, ICO, PNG, WEBP, and BMP.
   d) The colour *mode* of the image. Observed image colour modes were: RGB, CMYK, LA, L, RGBA, 1, P, and I.
4. The URL of the *first party*, i.e., the website the crawler visited when the resource was requested.
5. The URL of the *triggering origin*, i.e., who triggered the request for the resource. This can be the first party or a third party, e.g., via a script which then requests the image resource.

Next we discuss how this information can be turned into a real-valued vector representation. Table 4.1 gives an overview of the features we used for the classifier. Numerical features, such as image size, or colour values, can be used directly. Binary features are encoded as 1 if true or present and 0 if false or absent. Categorical variables can be encoded with one-hot vectors. A one-hot vector is a sparse vector with exactly one entry set to 1 and all others set to zero. We found, for example, 9 different image formats. If we use the order in which they are listed above the one-hot vector 100 000 000 encodes the information that the image is a GIF. For features with a potentially very large number of categories, such as netlocs, we constructed a ranking by counting the number of occurrences in all the data collected by the crawler and only used the top 500 for the encoding of the vector.

A URL can contain multiple different query parameters. We again ranked each query parameter according to how often it was found and used the top 500. We then encoded presence or absence of these top 500 query parameters for each URL resulting in a multiple-hot vector. The same approach was followed for path pieces. For header fields we only considered the four which were not found in all samples for the encoding.

Unique identifiers can be used to track users. They can be created using pseudo-random string generators. Such strings usually have high entropy,

Table 4.1: Pixel Feature Overview. "Is blocked" is only used in model 2, see also 4.5

Feature Name	Size	Description
Top Netlocs	500	One-hot vector of the most common netlocs
Top Query	500	Multiple-hot vector of the most common query parameters
Top Path Pieces	500	Multiple-hot vector of the most common 'words' in the path
URL Length	2	Number of character in path and in query part of URL
Has Query	1	Binary indicator, true if the URL contains any query
Query Count	1	Number of query parameters the URL contains
Entropy URL	1	Shannon entropy of the URL
Compressed URL	2	Size and reduction of URL after *zlib* compression
Is blocked	2	Whether it would be blocked by EasyList or EasyPrivacy
Header Fields	4	Binary indicator of presence of a specific header field
Header Length	1	Number of characters in the headers
Entropy Headers	1	Shannon entropy over all headers
Compressed Headers	2	Size and reduction of headers after *zlib* compression
Image Size	2	Width and height of the image in pixels
Is 1x1	1	Binary indicator, true if the image is of size 1x1 pixel
Image Format	9	One-hot vector indicating the format of the image, e.g., GIF
Image Mode	8	One-hot vector indicating the colour mode of the image
Transparency	1	Value of alpha channel of first pixel in RGBA
Image Colour	3	Colour value of the first pixel in RGBA
Top T.O. Netlocs	500	One-hot vector of the most common triggering origin netlocs
Third Party	1	Binary indicator, true if the pixel originates from a third party

while natural language terms not used to identify users tend to have low entropy. By measuring the entropy of, for example, a URL we can assess whether it might contain a unique identifier. We measure entropy in two ways. We directly compute *Shannon entropy*. High entropy data can often not be compressed well. We can therefore also approximate entropy by comparing the size of a string (e.g., the URL) before and after compression with the *zlib* library.

4.2 Classifier

The goal of training a classifier is to detect features and hidden patterns in the data that can be used to predict the purpose of a pixel correctly. Such a classifier could later be used in a browser extension like CookieBlock to enforce users consent choices, or more generally to improve our understanding of tracking pixels. This will be discussed in the next chapter. Here we will focus on the choice and training of the classifier.

Our task is a multi-class classification problem, as we want to predict one of four labels for each sample. It is supervised learning as we have labeled training data. The labels, or ground truth, are the labels we extracted from the consent notices or the label "necessary" we gave to all unmatched samples. Each sample is a sparse vector as explained above.

We use the XGBoost algorithm developed by Chen and Guestrin as it is currently among the most powerful classification technologies [6]. In particular, it achieved high performance in the similar study of cookie classification by Bollinger et al. [4]. It is optimised to deal with sparse data [6]. A practical advantage is that it is very fast, and scales well to thousands of features [6], and can therefore be run within minutes on consumer grade hardware [2, sec. 4.3 and 6.3.3].

XGBoost is a gradient tree boosting algorithm. In a multi-class classification problem it creates an ensemble (or forest) of decision trees for each class. A tree ensemble model cannot be optimised using traditional optimisation methods, hence the model is trained additively. Each boost round greedily adds the new tree, which most improves the model [6].

We first performed random search to find a good combination of hyperparameters. The chosen hyper-parameters can be found in the appendix A.3. We then performed 5-fold cross-validation to assess model performance. Each model was trained for a maximum of 300 boost rounds with early stopping after 20 rounds of no improvement of the validation score. We used the multi-class cross-entropy loss as validation score for the early stopping.

We trained two different models. Model 1 contains all the features listed in table 4.1 except "is blocked". We trained this model on 80% of the data, and will later use its predictions on the remaining 20% to compare our approach to filter lists in section 5.1. Model 2 contains all the features. It is used to investigate whether the decision by filter lists will be used as an important feature by the model.

4.3 Other Considerations

4.3.1 Class Imbalance

We observed significant imbalance between the number of training samples per classes. As shown in table 3.5, 84.5% of the samples are in the "necessary" class. In contrast, less then one percent of the samples are from the "functional" class. If ignored, class imbalance can lead to a low-quality classifier, and even worse, it might obscure the fact that the classifier is not performing well. For example, as "necessary" is so heavily over-represented, the classifier could achieve a high accuracy by mainly classifying these samples correctly. Even if it has a low accuracy for the other three categories the overall accuracy might be high, and thus not representing the real performance of the classifier.

One way to mitigate the effect of class imbalance is to multiply the loss for each sample with the inverse class weight of the predicted label. This is the approach followed by Bollinger [2, sec. 4.4.1], who also encountered class

imbalance. However, as our "functional" class contains even fewer samples, it is possible to have no samples in this class, particularly if the webcrawl was not run over many webpages or if we split the dataset into training and evaluation sets. Therefore we slightly adjusted the inverse class weight for a class j to avoid division by zero compared to [2].

$$w_j = \frac{\text{total number of samples}}{(\text{number of samples in class } j) + 1}$$

This approach will give under-represented classes a higher priority. However, it can not eliminate all effects of class imbalance [2].

4.3.2 A Note on the Impact of Misclassification

As we have seen in the feature engineering step a pixel can be represented as a sparse vector. Given such a sparse vector the model will produce probabilities for each pixel category. The discrete label for the sample can then be derived using a Bayesian Decision function. In this study we use *argmax*, i.e., choosing the purpose category with the highest probability. This approach implies giving each misclassification the same cost. In other words it is equally bad to classify a "necessary" pixel as "functional", than it is to classify it as "advertising".

If one wanted to account for the increasing privacy risk over the purpose categories from "necessary" to "advertising", one could define a cost function, where the cost increases with increasing distance between the true and predicted category [2, sec. 4.4.2].

4.3.3 Noise

Noise gives an indication of the maximum accuracy a classifier can achieve for a given dataset without overfitting. As mentioned above in chapter 3, one concern with sampling category labels for pixels from consent notices is that there might be intentional or accidental mislabeling leading to noise in the dataset. If there is too much noise it would seriously hamper any training of a classifier. We therefore estimated the noise in our dataset by assessing whether there are conflicting purpose declarations from different consent notices for the same pixel type. We considered something to be the same pixel type if its URL has the same standardised domain (i.e., `www.domain_a.com` is the same as `https:\\domain_a.com`) and path. Or put differently we define a pixel type by having a unique domain and path. An example for a conflicting declaration would be if on website A a pixel with path `/tr/` and domain `facebook.com` is labeled as "advertising", while on website B a pixel with the same path and domain is labeled as "necessary".

Table 4.2: Number and percentage of pixels labeled as part of that category but with deviating majority class considering only samples matched to a consent notice.

Label	Nr. of Samples	Nr. of Deviations	Percentage Deviation
Necessary	3950	1373	34.8%
Functional	108	67	62.0%
Analytics	39'790	805	2.0%
Advertising	85'976	1033	1.2%

Noise in Consent Declarations matched to Observed Pixels

Only looking at the samples which result from matching to a consent notice (129'824 training samples) we find 3833 different pixel types. Of these, 3337 (87%) were found in more than one consent notice. For every pixel type we counted how often it is declared in each of the four categories and considered the category with the highest number of declarations as the majority class for that pixel. For all pixel types which were found on more than one website we then counted how often declarations deviating from the majority class occurred. We found 3278 such deviations. The ratio between number of deviations and total training samples gives an estimate for the lower bound of noise across all training samples of 2.5%. When considering only the 129'328 matches from the 3337 pixel types which were found on more than one website and are therefore likely third party pixels, we get the same lower bound for the noise. We therefore found slightly lower noise among third party pixels than Bollinger et al. found for third party cookies [4] but similar to what they report over all samples [2, sec. 6.1.4].

The deviations from majority class are, however, not equally distributed across categories, see table 4.2. Only two percent or less of the training samples labeled as "analytics" or "advertising" are from pixel types whose majority class contradicts their label. One third of the training samples labeled as "necessary" and almost two thirds of training samples labeled as "functional" are from pixel types whose majority class contradicts their label.

Noise in the Training Data Set

Again we calculated the majority class per unique URL path and domain combination. We still refer to a unique URL path and domain combination as a pixel type to make comparison easier, however, in the full dataset many of the URLs will be for images. We found 430'653 different pixel types, 55'744 of which are likely third-party, as they were found on more than one webpage. We could match 438'253 observed HTTP requests to a third-party pixel type. Of these, 11'825 matches were to a purpose deviating from majority class. This results in a lower bound for the noise in third-party pixels and images

Table 4.3: Number and percentage of pixels labeled as part of that category but with deviating majority class considering all training samples.

Label	Nr. of Samples	Nr. of Deviations	Percentage Deviation
Necessary	619'569	6868	1.1%
Functional	49	14	28.6%
Analytics	63'043	768	1.2%
Advertising	130'501	4175	3.2%

of 2.7% and across all samples of 1.5%. Table 4.3 gives an overview of the noise distribution across the classes.

4.4 XGBoost Results Model 1

We start this section with an overview of the performance of the classifier. As discussed above, model 1, the main model, contains all the features we extracted from the consent declarations and observed pixels, but not whether EasyPrivacy or EasyList would block the observed pixel. After this assessment of the model, we use the model to gain more insights into misclassifications and feature importance. XGBoost reports different measures of importance, gain and weight, which will be discussed in the later part of this section.

4.4.1 Classifier Performance

There are different indicators of how well a classifier performs. Average accuracy measures the average proportion of correctly classified samples. Mean balanced accuracy, or macro recall, takes class imbalance into account. It is a more reliable measure for classifier performance if there is large class imbalance, as is the case in our dataset. Over five folds of cross-validation we reported an average accuracy of $98.77 \pm 0.03\%$ and a mean balanced accuracy of $95.84 \pm 4.20\%$.

Intuitively, a classifier should predict for every sample in a class that belongs to this class. In addition, all samples predicted to be in a specific class should also have this class as ground truth. The first is referred to as recall. It measures how many samples with ground truth b were assigned the correct class label b. The second indicator is precision. Precision of a class b measures how many samples that were assigned the label class b have the ground truth label b. Table 4.4 gives an overview of the mean and standard deviation of the precision and recall over five folds of cross-validation.

The lower mean and large deviation for both precision and recall for the "functional" category is likely partially due to the very small sample size.

Table 4.4: Mean and standard deviation of precision and recall for XGBoost model 1 using 5-fold split.

	Precision (Mean ± Stdev)	Recall (Mean ± Stdev)
Necessary	99.9 ± 0.0%	98.6 ± 0.0%
Functional	81.4 ± 5.4%	86.0 ±16.7%
Analytics	97.6 ± 0.1%	99.5 ± 0.2%
Advertising	94.4 ± 0.2%	99.3 ± 0.1%

Bollinger [2, sec.6.3.3] also found the highest deviations for the "functional" class, which is also the smallest class in his dataset. However, there are other problems with the "functional" category, that will be discussed in section 5.2.1.

Given the observed noise in our dataset (see section 4.3.3), there is an indication, that the classifier might be overfitting. It seems likely that it is just learning the samples in the "functional" category. Given that 28.6% of samples from the "functional" category have their majority class somewhere else, and manual inspection revealed that only very few might be truly "functional" (see section 5.2.1), it is perhaps not surprising that the model will have to memorize the "functional" samples.

4.4.2 What can we Learn from Misclassified Samples?

A confusion matrix can help to assess where the misclassifications happen. The confusion matrix lists for each true category i in row i how many samples are predicted by the model as belonging to class j in column j. The diagonal of a confusion matrix represents the accurate predictions for each class. Figure 4.1 shows the confusion matrix for the validation set and model 1. Figure 4.2 is based on the same data but shows in row i column j the fraction of samples from ground truth class i that is predicted as class j. On the diagonal we therefore find the recall. The high value of 1.0 for the "functional" class is, however, not representative, as we found that this class suffers from high variance across different folds of cross-validation, see table 4.4.

We will discuss two types of relevant misclassifications: ground truth "necessary" samples predicted as "advertising" or "analytics", and ground truth "analytics" or "advertising" samples predicted as "necessary". It is important to keep in mind that what we refer to as ground truth are the labels collected from consent notices or the label "necessary" for the unmatched samples that were not from one of the nine very common trackers listed in A.2, see section 3.4.1 for details. The labels are not necessarily the true purpose of a pixel. To analyse the misclassifications we chose 20 random samples from different pixel types and assessed whether their prediction by the model

might actually be more accurate than the ground truth label given to the sample. If that is not the case, we try to infer why the sample might be misclassified.

"Necessary" Samples Predicted as "Advertising" or "Analytics"

17 of the 20 pixels analysed with ground truth "necessary" and prediction "advertising" or "analytics" should most likely be in the latter classes. All were of size 1x1 and all would be blocked by the filter lists EasyPrivacy and EasyList. Both are indications for a tracking pixel (see 5.3). For 14 pixel types all or a very clear majority of consent notice declarations we found on other websites declare their purpose as "analytics" or "advertising", respectively. For three more the majority purpose is still "analytics" or "advertising", but the majority is less clear. One of them is the only sample that was actually matched to a consent declaration. The others are the result of our rather crude approach of putting every unmatched sample into the "necessary" category. This shows that the model could be very useful to flag tracking pixels, that were wrongly included in the "necessary" category.

The remaining three pixels are also all of size 1x1 pixel, and two of them would be blocked by the filter lists. For one of them we found one purpose declaration on other websites for "advertising" and one "undeclassified". For another we found two similar pixels with assigned purpose "advertising" but they had a slightly different netloc. This is also the one pixel which would not be blocked by filter lists. For the last pixel we did not find any consent declaration on any of the websites. We also did not find any consent declarations for other trackers by the website where we found that pixel. Whether these three pixels should be classified as "analytics" or "advertising" is less conclusive than for the 17 above. These three pixels might not have been found as easily as suspects of being wrongly placed in the "necessary" category without the model. We therefore think that this approach of training a classifier on labels from consent notices could be very helpful to then find less common undeclared tracking pixels.

"Advertising" or "Analytics" Samples Predicted as "Necessary"

All 20 samples with ground truth "advertising" or "analytics" predicted as necessary are most likely truly misclassifications, and should remain in their ground truth class. They are all of size 1x1, and would be blocked by the filter lists EasyPrivacy and EasyList, both indications for tracking. For each we investigated whether other websites declared a different purpose for it in their consent notice. All except two are always or with very few exceptions classified as "advertising" or "analytics", respectively. One other has a majority for "advertising". The last one has no clear majority.

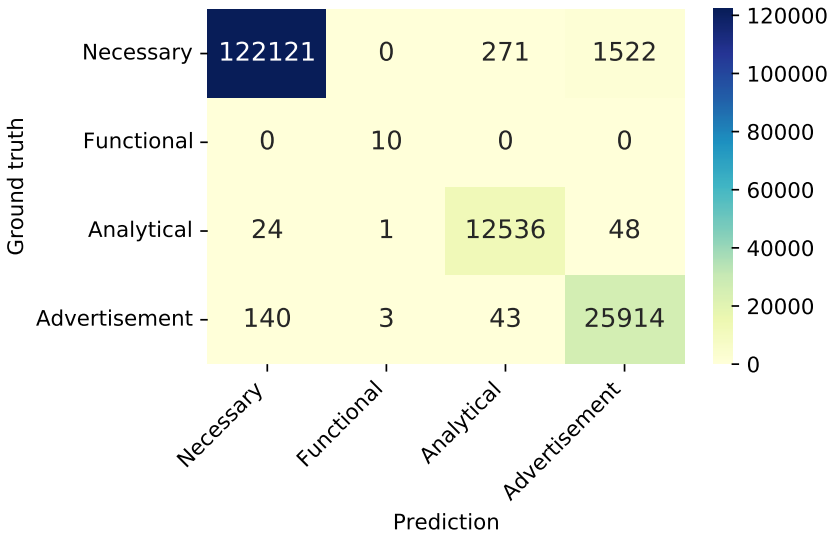

Figure 4.1: Confusion matrix for validation set. Rows represent the purpose assigned by consent notices (ground truth), columns the prediction from model 1. The value in row i and column j is the number of samples from ground truth class i predicted as class j.

The most likely reason for their misclassification as necessary is that the same pixel type could often not be matched to a consent declaration and was therefore added to the "necessary" category. This resulted in there being more samples for that pixel type in the "necessary" category than in the correct "advertising" or "analytics" category. We did count the total number of samples per pixel type and the number put into "necessary" because they could not be matched. It is the case that most of 20 pixel types analysed here had a majority of samples added to the "necessary" category.

These misclassifications therefore point at a weakness of our sampling approach more than at a failure of the model. They could, however, be used to find out which samples one might have to manually reclassify to improve the models accuracy in the real world.

Interestingly, all the samples have relatively short URLs compared to other tracking pixels. Three have particularly short URLs, one even has no query, no random string, not even the referrer and header fields are very different from what we see in "normal" images. It is, however, blocked by EasyPrivacy by two rules and clearly declared as "advertising". Probably this domain also has tracking pixels, the one in our sample can, however, not track a user. The model is therefore correct in predicting it as "necessary". For the other samples the short URLs and lack of long random strings might be the reason the model misclassified them. As we will see in the next section, length and entropy of the URL are important features for the classifier.

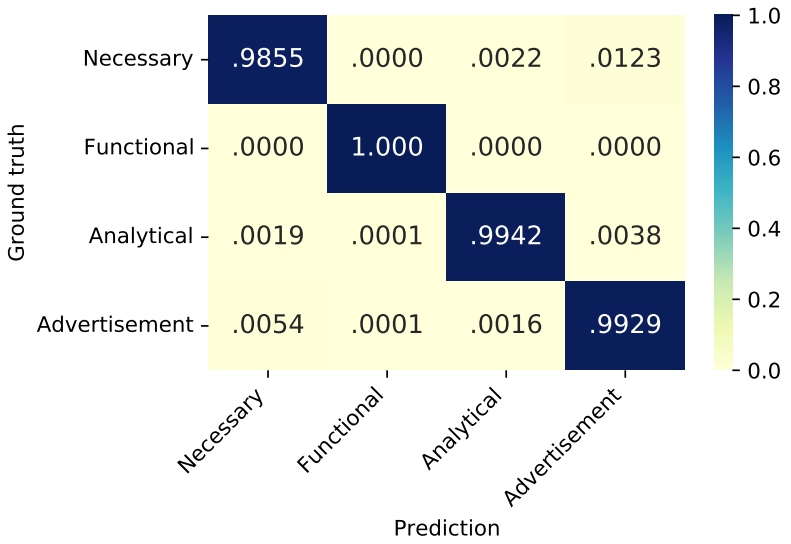

Figure 4.2: Confusion matrix heat map for the validation set. Rows represent the purpose assigned by consent notices (ground truth), columns the prediction from model 1. The value in row i and column j is the fraction of samples from ground truth class i predicted as class j.

4.4.3 Feature Importance

Feature Importance by Gain

Gain gives an indication of how much the expressiveness and decision power of the model increases when splitting on a feature. Or put differently, it is the average reduction in training loss gained when using that feature. Features with high gains therefore increase the model's ability to distinguish between the four pixel purpose categories more than features with low gain. Figure 4.3 shows the top twenty features ranked by gain. The average gain of splits that use a feature is given by the x-axis. The y-axis list the features with their associated vector index. For example top_path_piece-4 is the word 'collect'. It is used by Google Analytics, LinkedIn and 52 others in the path of tracking pixels. Most of the top 20 features are presence or absence of query parameters, path pieces and to a lesser extent netlocs.

Bollinger [2, sec. 6.2.1] observed that name and domain were among the most important features by gain for cookies. He voiced the concern that these features might be too specific and may cause overfitting. The same issue might occur with our query parameters, path pieces and particularly netloc features. We checked for the path pieces and query parameters in the top 20 features whether they are used in URLs from different netlocs and by how many. The heuristics is that if they are used in URLs from multiple domains, they are more likely to be good general features. For the four path pieces, we found that they are used by at least 45 different netlocs. Query parameter

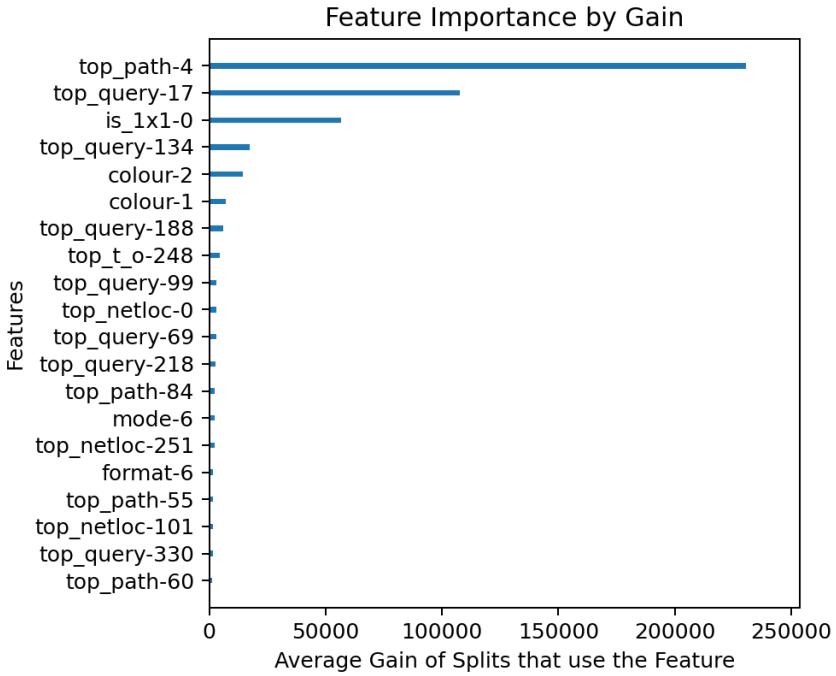

Figure 4.3: Importance by gain of the top 20 features for model 1.

top_query-218 is used by 81 netlocs, however, they are all from the same domain and contain random strings. This feature therefore is very specific and may cause overfitting. Other query parameters were found in URLs from 6 to 560 different netlocs. For example top_query-69 'h' is used by 560 netlocs, top_query-99 's' is used by 274 netlocs. This high number is probably partly due to the fact that it is statistically likely that different website operators chose the same one letter string as a name for a query parameter. Whether the model is overfitting when using a certain query parameter or path piece, might therefore be very much dependent on the query parameter or path piece in question.

It is not surprising that size, and particularly whether something is of size 1x1 pixels, is a strong feature. Given most samples in "necessary" are from images larger than 1x1 it should at least help to distinguish between this category and the others. Other important generic features are colour, format-6, which is whether the picture is PNG or not, and colour mode-6, which is P. We will investigate these features further in section 5.3.

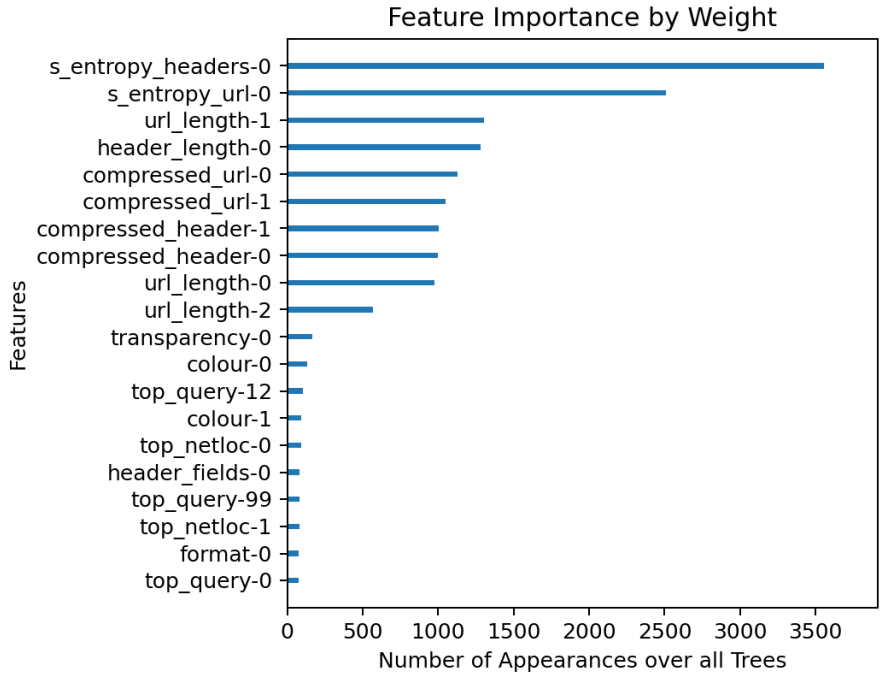

Figure 4.4: Importance by weight of the top 20 features for model 1.

Feature Importance by Weight

The feature weight denotes the total number of nodes in all trees that use this feature. Large weight means the classifier training relies heavily on this feature, as a high number of nodes use it. Figure 4.4 shows the top twenty features ranked by weight. The x-axis indicates how often a feature appears over all trees, the y-axis lists the features with their associated vector index. In general, many generic features, such as length or entropy, are important by weight. Many nodes use the Shannon entropy of the headers or URL, or 'compressed url/ header' which is also a measure for entropy (see section 4.1 for details). The length of the headers, path (url_length-0) or query (url_length-1) as well as the number of query parameters (url_length-2) can all be a measure on how much information the URL or HTTP request can contain. This will be further discussed in section 5.3.

4.4.4 Comparing Feature Importance by Gain and Weight

Only three features occur in both the top 20 features by gain and weight: top_netloc-0 which is used by Google Analytics, top_query-99 's' which is used in URLs from 274 different netlocs, and colour-1, which is the RGBA value for green. As the two metrics measure different aspects of feature

importance the fact that we do not observe more overlap is not that surprising. For a feature to have high weight, it has to be used in many splits over all trees. On the other hand, a high gain indicates that when splitting on a feature the decision power of the model increases a lot, but it does not necessarily mean, that the split is used often.

A high gain, lower weight feature could be a feature close to the root of the decision trees. If, for example, an image is of size 1x1 pixel, it is probably not a normal image. This feature could therefore be used as or close to the root in the trees of the "necessary" category to exclude most tracking pixels from this class. It would therefore increase the decision power of the model significantly. The feature might, however, only be used once per tree in a few trees, giving it low weight. We did indeed observe the feature is_1x1 as the root of many trees for the "necessary" category.

A low gain, high weight feature could be a feature that is used in many splits close to the leaves. Each of these splits on its own might not be very important for the model, i.e., it may do little to increase the decision power of the model. We did observe the features connected to entropy and length of the URL and headers to be found all over the decision trees we inspected, including close to the leaves.

4.5 XGBoost Results Model 2

As the only difference between model 1 and 2 is the use of the feature "is blocked" by model 2, we will proceed in the same way as in 4.4 and focus on feature importance by gain and weight. The top end of the top twenty features changes in order slightly between model 1 and 2, but the same features remain important. Both by gain, see figure 4.5, and weight, see figure 4.6; both the feature of being blocked by EasyPrivacy (is_blocked-0) or EasyList (is_blocked-1) are among the top twenty features. The information whether a pixel would be blocked is therefore clearly useful to the model to distinguish between the four purpose categories.

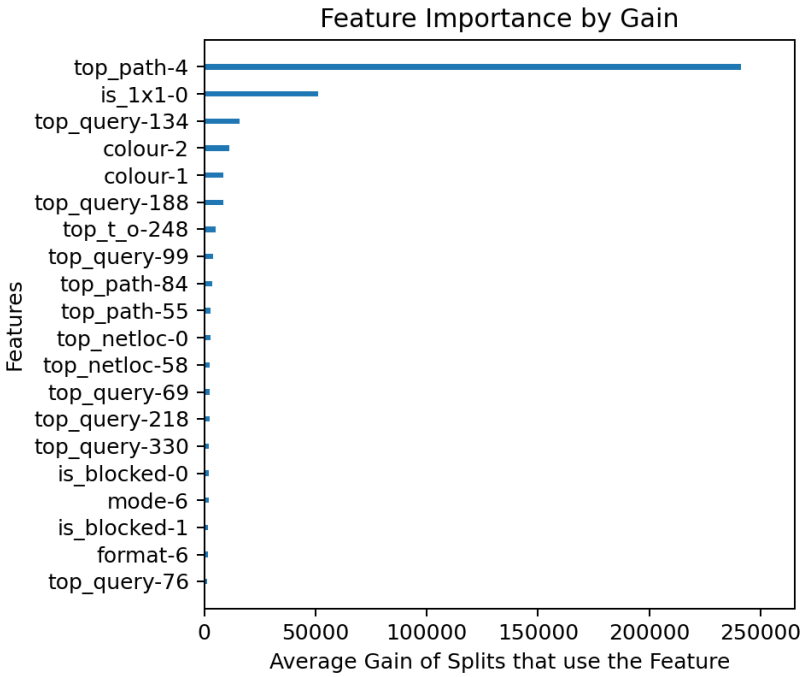

Figure 4.5: Importance by gain of the top 20 features for model 2.

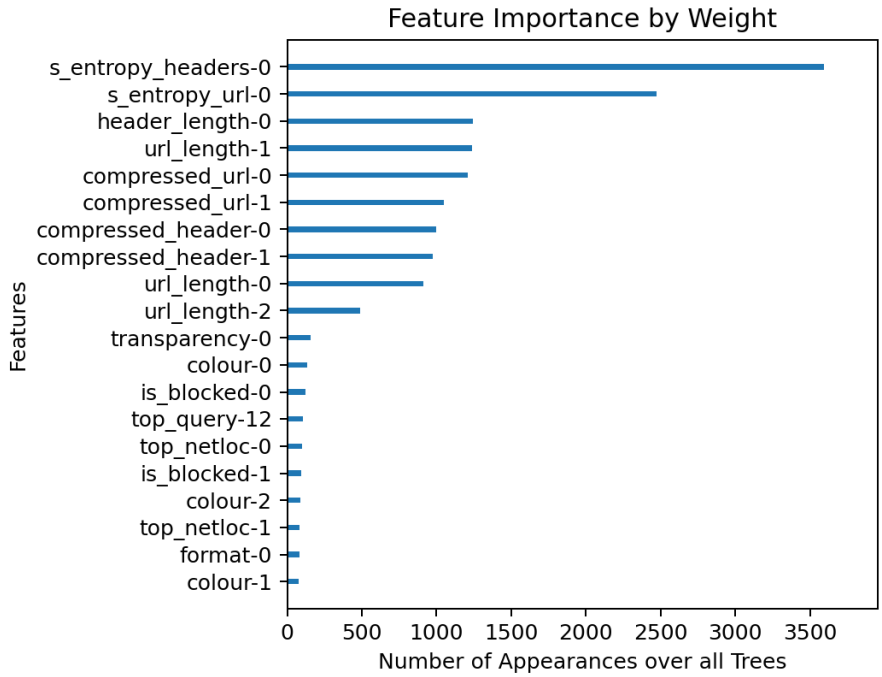

Figure 4.6: Importance by weight of the top 20 features for model 2.

Chapter 5

Making the Pixels Visible

In this chapter we use the dataset collected by the webcrawl and the classifier trained on it to analyse different aspects of tracking pixels. First we compare the assessment of privacy threat as indicated by the purpose declared in consent declarations and the judgment of filter lists. We follow this by a discussion on whether the purpose categories are meaningful for pixels and give heuristics on how to recognise a tracking pixel in the wild. We end the chapter with notes on GDPR compliance.

5.1 Comparing Filter Lists and Consent declarations

Filter lists provide an orthogonal view on tracking to consent notices. While it is in the interest of actors in the tracking and advertising industry to collect as much data as possible while staying compliant, filter lists are a tool in the hands of the users to protect their privacy and avoid seeing ads. The objectives of the EasyPrivacy filter list for example state that it must remove tracking [12] (See also 2). We therefore expect that filter lists, particularly EasyPrivacy, would block all "advertising" and "analytics" pixels, and would not block "necessary" and "functional" pixels. As both the purpose categories and the decision to block or not to block are assessments of the potential privacy threat of a pixel or image, it makes for interesting insights to compare the two views.

With a corresponding parsing library[1], filter lists can be used offline to query whether a URL should be blocked. We consider an image or pixel as blocked if at least one of the filter lists, EasyList or EasyPrivacy [12], would block it. The filter rules often depend on resource type and whether it is third-party. We therefore set resource type to 'image' for all our queries to the filter lists and assessed per sample whether it was third party.

[1]We used https://github.com/englehardt/abp-blocklist-parser, last accessed 13.02.2022

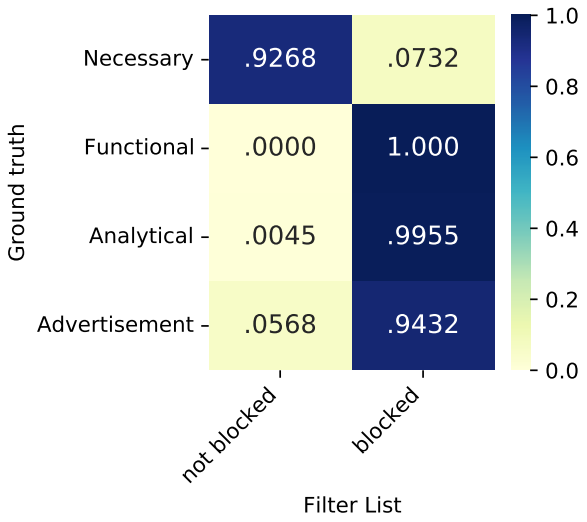

Figure 5.1: Comparing consent declarations and filter list decisions. Rows represent the purpose assigned by consent notices (ground truth), columns the decision of filter lists to block or not. The values in row i are the fractions of samples from purpose category i that are blocked or not by the filter lists EasyPrivacy and EasyList. For the actual numbers of samples per category see A.1.

We compared the decision of the filter lists with the purpose declared in consent notices, see figure 5.1 and with the purpose predicted by our model 1 for the validation dataset, see figure 5.2. Both figures show the purpose categories on the y-axis and the decision by the filter lists on the x-axis. The values are the fraction of samples in that purpose category that are blocked or not by the filter lists. Overall, both purpose as declared in the consent notice or as predicted by the model, and the decisions by the filter lists agree in their assessment of privacy threat. The majority of "necessary" pixels and images are not blocked, while the majority of "advertising" or "analytics" pixels are blocked. The interesting cases are where they do not agree.

"Analytics" or "Advertising" samples not Blocked by Filter Lists

We analysed 20 random samples of different pixel types with ground truth "Analytics" or "advertising" that are not blocked. Two are not pixels but images and the result of a mismatch to a consent declaration on the one hand and being moved from "necessary" to "analytics" wrongly on the other due to having the same domain as one of the common trackers we used to reclassify. They therefore mainly point at some of the weaknesses of our approach.

All other samples are of size 1x1, which is indicative of them likely being tracking pixels, see also section 5.3. For 12 samples we found that all or a

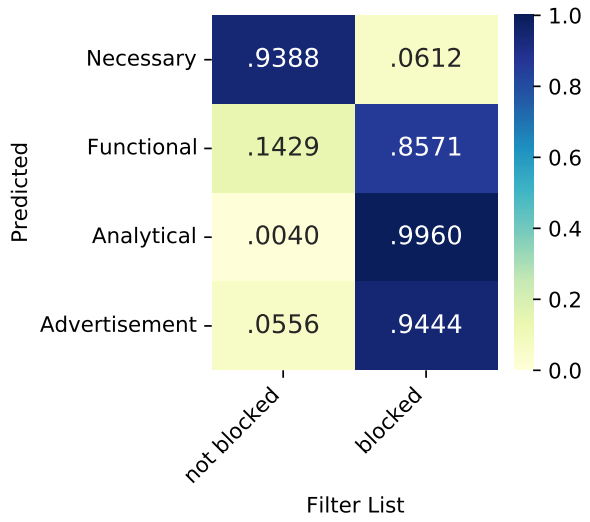

Figure 5.2: Comparing model predictions and filter list decisions. Rows represent the purpose category predicted by model 1, columns the decision of filter lists to block or not. The values in row i are the fractions of samples from predicted purpose category i that are blocked or not by the filter lists EasyPrivacy and EasyList. For the actual numbers of samples per category see A.2.

huge majority of consent declarations from the websites in our dataset agree on their purpose being "analytics" or "advertising", respectively. A possible explanation why these samples are not blocked, even though they come from what seems like relatively well known trackers, may be that the pixels are loaded by a script that itself is blocked by the filter lists. Consequently, they would not need to also block the pixel to stop the tracking, however, we did not investigate this further. Interestingly, two of the samples, however, even though having all consent declarations agreeing on being "advertising" are not predicted as such.

Another three samples are only declared and found on one website. Two of the samples are first party pixels, and are only declared in the consent notice of the website they came from. Another pixel was only declared and found on one website, although not being first party. It is not surprising that such rare and particularly first party pixels do not get blocked by filter lists based on their domain. Rare pixels might of course get blocked based on more general rules. If, for example, they would have named their tracking pixel "track.gif" it would get blocked. As Bollinger et al. [4] discussed for cookies, the approach of collecting purpose labels from consent notices enables them to collect information on unique cookies from the host that created them. They discuss this as one advantage of their approach over using a database such as the *Cookiepedia* repository [22], which might not contain rare cookies. Similar considerations might apply here. The EasyList filter lists rely on human operators to add the rules. Rare pixels might have not (yet) come to

their notice. Using our data collection from consent notices may help privacy list maintainers to find rare pixels and keep their lists up to date.

For the last three samples consent notices from different websites state different purposes without clear majority or with many unclassified samples. They might therefore not truly be disagreements in assessment of privacy threat between filter lists and purpose declarations. It might be that the wrong purpose was declared in the consent declaration.

Of the 1540 samples with ground truth "analytics" or "advertising", 1525 are also predicted by the model to belong to that class. An additional 56 samples are predicted as "analytics" or "advertising", with most being of the same pixel type as the samples in the ground truth set, but assigned another purpose in the consent declaration. The similarity between ground truth and prediction for the "analytics" or "advertising" class can also be inferred from the fact that the recall for both classes is over 99%, see table 4.4, and that there are therefore almost no false positives. We therefore refrain from analysing the 'predicted "analytics" or "advertising" not blocked' dataset in depth, due to its similarity, and instead focus next on samples with ground truth "necessary" that would be blocked by filter lists.

"Necessary" Samples Blocked by Filter Lists

We again analysed random samples from 20 different pixel types. Table 5.1 gives an overview. Four were larger than 1x1 pixel. Two of them seem to be actual non-tracking images. One is blocked by a relatively general rule and is an advert banner (sample id 1295_190). The other one is blocked because of its domain (sample id 6543_597). We do indeed also find declarations for "advertising" pixels in our collection of consent declarations for that domain. However, there is no indication that this image is doing any tracking and it looks like abstract art and there is no text in the image. It also contains the words 'kunstveiling' (art auction) in its URL, which also hints at it being art and not an advert. Changing parts of its URL that seem random does fail to retrieve the image, which can be used as a heuristic that it is not doing any tracking, see section 5.3. Both large images are blocked by EasyList. As declared in their policy, the purpose of the EasyList filter list is that it must remove adverts and may remove trackers [12]. It is therefore expected that it would block an advert such as the first image. The second large image, however, should not be blocked, the blocking rule based on the domain alone might be too general. As a normal image, this is correctly put in the purpose category "necessary".

The third image is 6x5 pixels large (sample id 8957_47). It comes from `https://pixel.wp.com/g.gif`[2]. It is blocked based on the domain and

[2]The query part of the URL was removed.

Table 5.1: Overview of the 20 "necessary" samples blocked by filter lists. Key for the 'purpose' column: adv = advertising, stat = statistics/ analytics, nec = necessary, uncl = unclassified or unclear (in all these cases we found consent declarations for the exact pixel). xA = for the same domain a different pixel is labeled as "advertising" (i.e., domain is likely a tracker). - = no consent declarations found.

Sample ID	Purpose	Blocked based on	Blocked by	Size
1295_190	-	part of netloc	EasyList	300x45
6543_597	-	domain	EasyList	1000x1397
8957_475	nec	domain, general	EasyPrivacy	6x5
7742_137	-	domain, path	EasyPrivacy	2x2
9376_435	adv	domain	EasyPrivacy	1x1
7542_130	stat	general	EasyPrivacy	1x1
7562_217	adv	domain	EasyList	1x1
9244_215	adv	general, path	EasyPrivacy	1x1
8598_313	stat	domain	EasyPrivacy	1x1
7293_278	adv	domain	EasyList	1x1
4737_1184	adv	part of netloc	EasyList	1x1
5738_452	adv	domain	EasyList	1x1
570_305	adv	domain	EasyList	1x1
3610_607	xA	domain	both	1x1
9148_205	xA	domain	EasyPrivacy	1x1
8274_303	xA	domain	EaysList	1x1
656_391	xA	domain	EasyList	1x1
2283_850	xA	domain	EasyPrivacy	1x1
6531_191	-	domain	EasyList	1x1
2499_231	uncl	part of netloc	EasyPrivacy	1x1

'.com/g.gif' by EasyPrivacy. For the same domain and netloc, we only find declarations for a pixel called 't.gif' with purpose "necessary" in our dataset. Many domains use multiple different pixels, sometimes for different purposes. It is possible that `pixel.wp.com` has a "necessary" pixel called 't.gif' and an another one called 'g.gif' used for advertising. However, 't.gif' would also be blocked by EasyPrivacy. We therefore expect that this sample is wrongly assigned the purpose category "necessary" by the consent declaration.

The fourth large sample is a 2x2 pixel[3] and will be treated together with the the remaining 16 samples that are 1x1 pixels. For nine pixel types all or a very clear majority of consent declarations give "analytics" or "advertising" as purpose ('adv' or 'stat' in table 5.1). For another five we did not find an exact match in any consent declaration but for the same domain and different pixels a clear majority are "analytics" or "advertising" (xA in table 5.1. While we do find declarations for these or similar pixels on other websites, these

[3]This is one of the Adobe pixels. It will be discussed further in section 5.3.

14 samples could not be matched to a consent declaration on the website where we found them. They were therefore all added by us to the "necessary" category. As with the case discussed above, where "necessary" samples are predicted as "analytics" or "advertising", this is not really a disagreement in privacy threat assessment between the filter lists and the purpose declared by consent notices. Instead, it points at a weakness of our approach, yet also hints at a solution. One could use the filter lists to help detect unmatched tracking pixels. These are all potential candidates to be in violation of GDPR compliance. Of course one would also have to verify that there truly is no consent declaration for a specific pixel on the webpage it was found on, and the matching did not just fail.

For the remaining two samples of size 1x1 we either found no declarations on any website or they were unclassified. Two were blocked based on their domain, one based on the first part of its netloc. For samples that 'look like pixels', see also section 5.3, but where we find no consent declarations, the filter lists may help us assess the privacy threat of a pixel.

Looking at why the 17 pixels get blocked revealed that 12 are blocked based on their domain, one on the domain and path, two on the first part of the netloc, and the last two by more general rules, such as path pieces or the name of the pixel. If a pixel is blocked based on its domain, we take this as a strong indication, that the domain is a well known tracker. Eight pixels are blocked by EasyPrivacy, one by both lists, and another eight by EasyList. This shows that it is definitely worthwhile considering different lists.

Conclusion

Comparing the assessment of privacy threat by filter lists and through the purpose declared in consent declarations reveals three main findings. First, taking both the decisions made by filter lists and a majority vote on purpose declared by the consent notices into account can help to confidently detect which of the unmatched samples are tracking pixels. Decisions by filter lists can also help where there are no declarations. Second, our approach might be able to detect some rare tracking pixels, that have been missed by filter lists. Third and most importantly, these two sources agree to a large extent. This also means that the functionality provided by filter lists is very similar to what we might have achieved if we would have incorporated tracking pixels into the browser extension CookieBlock [4] to automatically enforce users GDPR consent preferences for pixels in addition to cookies. Another reason why adding pixels to CookieBlock might not make sense is that the categories "necessary" and "functional" have issues, which will be discussed in the next section.

5.2 Are the Purpose Categories Meaningful for Pixels?

As mentioned in section 3.1.3 the CMP Cookiebot uses the same categories as we do, indicating that they must see sense in it. However, they use the categories for all sorts of trackers, including cookies. It might therefore still be the case, that the categories make perfect sense for cookies, but are less ideal for pixels. This seems likely as cookies also outnumber pixels in the consent declarations, see table 3.3, making it likely that a shared category scheme might be optimised for them and not pixels. Our concerns with the categories stem mainly from the fact that the two categories "necessary" and "functional" have fewer samples that could be matched to a consent declaration and that there is high noise for those categories, see section 4.3.3. The "functional" category is also not well predicted by the model. We therefore analysed all samples that where matched to the purpose "functional" in more detail, and will present a short discussion of the "necessary" tracking pixels. We limit the discussion for the "necessary" to potential pixels, i.e., everything of size 1x1 as this is a reasonable indication, that something is a pixel, see section 5.3.

5.2.1 Are there truly Functional Pixels?

We collected 57 consent declarations for "functional" pixels, as shown in table 3.4. Some of these declarations are for the same pixel type (e.g., `facebook.com/tr`) but were collected from different websites. If we only count the different pixel types declared, we found 49. This very low number compared to the other categories and the much higher noise, see table 4.2, raised the question, whether "functional" is a meaningful category for pixels.

Of the 49 declared pixel types we could match 16 to the observed requests for image resources. This resulted in a total of 108 samples in the "functional" category [4] as one declaration can be matched to multiple observed pixels. We assessed whether they are likely to be truly functional in the following ways.

1. Is the same pixel declared in a different category on other websites? For this we queried the consent_data table and counted how often a name, domain pair is declared in each category.
2. Is there (easy) online evidence that it is a tracker? For this we searched online for the domain given in the declaration.
3. Is it blocked by filter lists? For this we queried the EasyPrivacy and EasyList using the BlockListParser library from [9].

Of the 108 samples in the "functional" category only pixels from two different domains did not get blocked by EasyPrivacy or EasyList. For one domain we

[4] In this section we discuss all the samples originally matched to a "functional" purpose declaration. This includes the common trackers listed in A.2, which we moved for all the other analysis.

found no other evidence that would contradict their being truly "functional". For the other domain we find purpose declarations for all purposes including many "unclassified". From the sparse information given in the consent notice and the contradicting purpose declarations it is not readily possible to ascertain its true purpose. We did not find any online evidence that is is a tracker.

Pixels from one other domain were blocked but by a general rule. The pixel is named "track.gif", which is flagged by EasyPrivacy, but the domain is not blocked. We found no further evidence that this pixel is used for analytics or advertising. Our outsider perspective does of course not allow us to say beyond doubt that these are truly "functional" pixels. It seems, however, that we found at least one, and at most three, different types of "functional" pixels.

We may not find as many "functional" pixels as declared because our webcrawler only interacts in a relatively simple way with the website. If "functional" pixels are, for example, mainly used for forms, or logins we would not collect them. As Dino Bollinger [2, sec. 6.1.3] previously pointed out, this is likely to specifically decrease the number of "functional" or "necessary" cookies observed. It is therefore also likely to decrease "functional" or "necessary" observed pixels. But given that the number of declared "functional" pixels was already much lower than any other category, see table 3.4, even if we observed every last "functional" pixel, their number would still be very low.

5.2.2 Are there truly Necessary Pixels?

As for the "functional" pixels, we found relatively few "necessary" pixels that we could match to consent declarations and there is a high amount of noise, see table 4.2. In addition, only 8.0% of the "necessary" samples matched to a declaration would not be blocked by the filter lists EasyPrivacy or EasyList. Similarily only 6.7% of the unmatched pixels of size 1x1, that we added to the "necessary" category, are not blocked by the filter list. This is a similar proportion at the sample level than what we observed for the "functional" category. It is also a good first indication, that truly necessary pixels are at least very rare compared to the other categories.

5.2.3 Conclusion

Even with our in depth analysis of all the samples labeled as "functional" our outsider view does not allow us to determine conclusively whether there are true functional pixels. For the "necessary" category we found between 6.7 and 8.0% of 1x1 pixels to be potential candidates for being truly necessary. It does seem likely that the categories do represent an aspect of real world

use of pixels, particularly also because our approach might undersample "functional" and "necessary" pixels. Pixels from these categories are, however, much rarer than "analytics" or "advertising" pixels. The same is also true for cookies [4].

5.3 How to Recognise a Tracking Pixel in the Wild

In this section, we discuss heuristics that can be used to recognise a tracking pixel, i.e., a pixel from the "analytics" and "advertising" categories, as such. On its own each one might not be conclusive, but taken together they can give a strong indication. The heuristics are based on anecdotal observations made during this study, and on what we learned from the model.

5.3.1 Size: Not all Tracking Pixels are 1x1, but most are

The feature "is_1x1" has a high gain in our model 1, see figure 4.3, indicating that whether an image is of size 1x1 is useful both to the model and to us to recognise something as a tracking pixel. The vast majority of tracking pixels in our dataset were indeed of size 1x1 pixels.

All 72 images that were matched to a declaration and were bigger than 1 pixel turned out to be most likely mismatches. As we saw in section 3.4.1 our way of matching consent declarations to observed pixels is not perfect. For example a declaration with the name `collect` and domain `fashion.com` will be correctly matched to a pixel with URL `http://fashion.com/collect/tracking_pixel.GIF` and wrongly matched to any image from a folder called `autumn-collection`, e.g., an image with URL `http://fashion.com/autumn-collection/purple_hat.PNG`.[5]

We did, however, find non-declared tracking pixels of size 2x2 pixels when looking at image resources that would be blocked by EasyPrivacy or EasyList. There were 522 images of size 2x2, which are blocked by the filter lists. Of these, 184 were from Adobe Analytics, which uses the domain `omtrdc.net` for their tracking pixels. Adobe Analytics also state in their implementation guidelines that their "mobile beacon image" as they call it, is 2x2 pixels large[6]. A further 160 2x2 images were from the domain `2o7.net` which is a known tracker. Quickly assessing the remaining 178 2x2 pixels using some of the heuristics presented in this section, reveals that most or all are likely candidates for tracking pixels. Of course also images of other sizes, such as 1x2 or 2x3 et cetera, could be used as tracking pixels. We did not investigate this further. However, our results clearly indicate that it is not sufficient to only consider 1x1 images when hunting for tracking pixels in the wild.

[5] This example is based on an observation from the real world, but simplified here. It accounts for 26 of the 72 "large tracking pixels".

[6] https://barrymann.files.wordpress.com/2014/03/oms_sc_implement.pdf

As mentioned in chapter 2 not every 1x1 pixel image is a tracking pixel. They could, for example, also be used to position elements on a webpage. We did not specifically investigate how common such other uses of 1x1 pixel images are. However, we feel confident, that if they were common, we would have encountered them. This might therefore in practise not be a big concern.

Size and particularly whether an image is of size 1x1 pixel is a good first indication for a tracking pixel. When sampling images from HTTP requests the size of an image can, however, only be inferred once the image is delivered, which is after the tracking information has reached the pixel server. It is therefore not a useful feature in any application that tries to recognise tracking pixels with the aim of preventing tracking.

5.3.2 High Entropy and Information

Many features with high weight, and therefore importance, found by the model are measures of entropy and information content of the URL 4.4. One possible explanation for this is that URLs can contain high entropy pseudo-random strings as unique identifiers to track users. By measuring the entropy of a URL, we can indirectly assess whether it might contain a unique identifier. In the wild this unique identifier can be found in the netloc, the path, the image name, the query parameters, or any combination thereof.

A real world example of a random netloc in the URL of a tracking pixel is `https://w2txo5aaxmi27hddahoncgxp2bwomwddvbnz5r7s6be29835369921bb-am1.e.aa.online-metrix.net/fp/clear.png?org_id=w2txo5aa&session_id=prsktdt1rhhv35ggtezs2jhl&nonce=6be29835369921bb&di=yes`. It is interesting that the first eight characters of the random netloc is equal to `org_id` in the query and the sixteen characters before `am1` is equal to `nonce` in the query. The `nonce` or the numbers in between `org_id` could likely be used to identify a user. Manually changing numbers in the browser still retrieves the same pixel.

It is also possible to have a random number identifier as an image name. For example, we found the URL `https://zwaluwhoeve.nl/pagespeed_static/1.-JiBnMqyl6S.gif`. Changing the image name string in the browser as long as it contains `1.x.gif`, where x stands for one or more letters or numbers, retrieves a pixel. Normal images can of course also have a random looking name, particularly if Universally Unique Identifiers (UUIDs) are used to name them. This may, for example, be the case if a server has to store a large number of images or if the images were generated automatically. In contrast to tracking pixels, changing any number in their name will result in either an error or another image being returned.

Another source of high entropy of a URL is if it contains query parameters with random identifiers. For example, Facebook's tracking pixel contains all

the tracking information in the query. Some Facebook pixel queries were over 1'600 characters long.

A good heuristic to recognize a tracking pixel is therefore to check whether the URL contains anything that 'looks like a random identifier', change it, and see if the same image or pixel is returned.

5.3.3 Get to know your Pixels: Domain and Path

We observed that a huge number of tracking pixels come from a relatively small number of companies. Recognising those from their domain or netloc helps to identify a large number of pixels in the wild. Words used in the path of the URL can also be used as clues. For example, in our model the word `collect` in the path is an important feature. It is used by Google Analytics, LinkedIn and 52 others in the path of their tracking pixels. One can therefore recognise a large number of tracking pixels by knowing the domains of common trackers and some keywords. This way of recognising pixels is conceptually similar to what filter lists do and is very effective.

As discussed in section 5.1, filter lists, such as EasyPrivacy, were developed with the aim of blocking unwanted trackers based on their URL. They therefore contain a set of rules that determine based on the domain and keywords like "track.gif" whether something is likely a tracker and should be blocked.

5.3.4 Why Transparency and Colour are not that useful

As tracking pixels are also referred to as "invisible pixels", one might be tempted to think they are always transparent. This can be achieved in multiple ways, including image transparency, CSS transparency, overlaying the pixel by other content (low Z-index), or using same color as the background. Unfortunately, our sampling approach only allows us to determine whether the original image was transparent, and not the other methods. We found that most "advertising" pixels are transparent and most "analytics" pixels are not. This, however, likely stems from the fact that most "advertising" pixels are from Facebook and most "analytics" pixels from Google Analytics, and one chose to make the original image transparent and the other did not. The same explanation is probably behind the fact, that we see a similar split between "analytics" pixels being mainly mainly white and "advertising" being mainly black. Only very rarely did we find a tracking pixel that was not either black or white. As with size the colour of the original pixel is only known after loading it, and therefore after the tracking happened.

5.4 Detecting Potential Violations of GDPR Compliance

Bollinger et al. [4] present a comprehensive collection of ways to detect potential violations of GDPR compliance for cookies. Many can be easily applied to tracking pixels. As finding undeclared tracking pixels is more challenging than finding undeclared cookies, we will elaborate on this in more detail and revisit our main results from the point of view of detecting potential violations of GDPR compliance.

5.4.1 Undeclared Tracking Pixels

Undeclared tracking pixels are tracking pixels present on a website but not declared in the consent notice. This means that the user was not informed and could not consent to the data collected through these pixels. The requirement of informed consent might therefore be violated.

Not all of the potential tracking pixels that could not be matched to a consent declaration, are violations of informed consent. As discussed in section 3.4.1, the way we match consent declarations to observed pixels is prone to errors. For example, if the name contained a typo, or a slightly different version of the name was pasted into the name field by the website operator, our script would fail to match the name to the observed pixel. To a human user the consent declaration might still clearly match the observed pixel and could be considered informed consent. One would therefore need to determine for every case whether it was truly not declared or we failed to match it.

Unmatched potential tracking pixels could serve as a start when investigating potential violations to consent. Unfortunately, there is no a priori way to distinguish with certainty between images and tracking pixels in the set of unmatched samples. We propose three different ways to find those unmatched pixels. The first uses the heuristics presented in section 5.3, the second takes advantage of the trained classifier and the third uses information obtained from filter lists. Of course, they can and probably should also be combined.

First, one could use the heuristics presented in section 5.3. Using only the heuristics of considering everything of size 1x1 as a potential tracking pixel, we found that 15.7% of images not matched to a declaration could be tracking pixels, see also section 3.4.2. Using more than one heuristic would of course improve this assessment and narrow down the number of samples to assess manually.

As a side note, we observed that 27'614 (25%) unmatched pixels were from `google.nl` and contained `pagead/1p-conversion`, `ad/ga-audiences` or `pagead/1p-user-list` in their path. We commonly find declarations not

for these, but for google.com. We also often observed pixels from both google.nl and google.com on the same website. Given there is no rule on how consent notices have to look, they could also use the company name instead of the domain. If that was the case it would be obvious that the two (.nl and .com) are the same company. Of course this is not a legal assessment, but it seems reasonable that this is not a violation as long as the corresponding google.com pixel was declared on the website the google.nl pixel was found on. If we exclude google.nl in the analysis above we get the lower estimate of 12% of tracking pixels in the the "necessary" category. They were found across 6519 different webpages (62.8% of pages visited).

A second approach to finding unmatched tracking pixels is to analyse the samples which were 'misclassified' by the model. Manually analysing samples with ground truth "necessary" predicted as "analytics" or "advertising", see section 4.4.2, revealed that the model was correct in its prediction in many cases. It even helped to find rare undeclared pixels that might not have otherwise been found.

The third way to detect tracking pixels, is to check the unmatched samples against privacy filter lists, see also section 5.1. This approach might, however, miss some rare pixels, which the model could find.

In figure 5.3 we give a comparison of what percentage of samples are mismatches, and which might be potential violations, according to each method. For the heuristics method it is the percentage of 1x1 pixels in all unmatched images excluding google.nl. For the model it is the percentage of samples with ground truth "necessary" that are predicted as "analytics" or "advertising". And for the method involving filter lists it is the percentage of samples with ground truth "necessary" that are blocked by at least one of the filter lists. One reason why the heuristic method finds a higher percentage of mismatches is that it was applied to the whole dataset before the nine common trackers, as listed in appendix A.2, were removed; the other two are after this.

5.4.2 Outlier from Majority Label

A relatively straight forward way to detect potential violations is by comparing purpose declarations across different websites. If a label for pixel x on website A deviates from the majority of labels for x by all other websites, it might be an indication that A assigned the wrong category to x, which might be a potential violation of GDPR compliance.

In section 4.3.3 we counted how many samples per category have labels that deviate from the majority class to asses how much noise our dataset contains. We found that the deviations from majority class, see table 4.2, are more common for pixels labeled "necessary" or "functional" (34.8% and

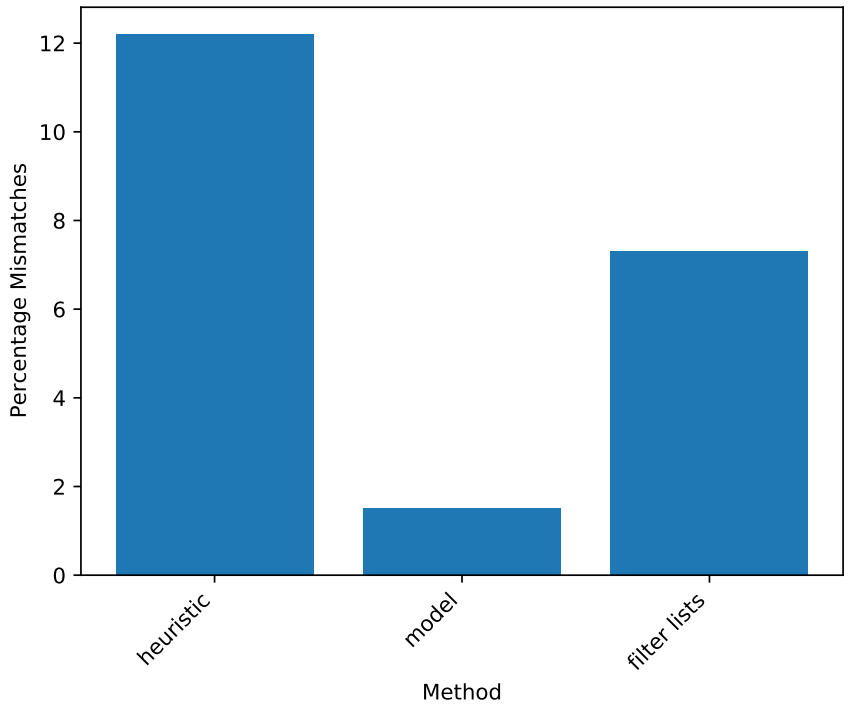

Figure 5.3: Mismatches which might be potential violations as percentage of samples found by each of the three methods.

62.0%, respectively) than for "analytics" or "advertising" (2.0% and 1.2%, respectively). Also anecdotal evidence suggests that many of the deviations from majority class for "analytics" or "advertising" are between each other, while many of the deviations from majority class for the "necessary" or "functional" category are between them and "analytics" or "advertising". It is particularly worrying in view of the protection of privacy if "analytics" or "advertising" pixels are labeled as "necessary", as this category does not require consent [10]. At the level of websites, we found pixels matched to a consent declaration with deviating majority class on 618 different webpages (5.9% of pages visited).

A limitation of this approach is that the majority class might not necessarily represent the true purpose. If this is the case we underestimate the number of potential violations. Another limitation is that it does not work well for rare pixels. And it does not help us detect wrongly labeled first party pixels or pixels which only occur on one website.

5.4.3 Wrong Label for Known Pixel

Google Analytics pixels are very common and well known. They are used for analytics purposes as the name suggests. For the equally common Google

Analytics cookie the EU Court of Justice ruled in the Planet49 case that it is not compliant to declare them as “strictly necessary” [17].

Of the 3’950 samples of tracking pixels in our dataset that were matched to a consent declaration stating as purpose “necessary”, 600 (15.2% of samples) were from Google Analytics. They are therefore very likely violations of GDPR compliance. These samples came from 73 different websites. For one other website we found a declaration for Google Analytics as “necessary” but did not find the pixel. Another 22’560 samples from Google Analytics could not be matched to a declaration. As discussed above, they might also be potential violations.

Chapter 6

Related Work

In this chapter we discuss related work that studied tracking pixels or automatic blocking of trackers and explain them in the context of this work.

Fouad and Bielova et al. [13, 1] proposed a new classification of tracking behaviour based on observations from invisible pixels. Like our study they collect tracking pixels from HTTP requests and responses found during a webcrawl using OpenWPM. They crawled the top 10'000 Alexa domains, collecting data from 84'658 webpages from 8'744 domains. They defined eight different categories of tracking behaviour from the observed behaviour of third party pixels . They then applied this classification to the full dataset, which in addition to pixel data also contains first and third party cookies. This allowed them to detect new categories of tracking, analyse the interplay of cookies and tracking pixels, and detect unknown collaborations between domains. Finally, they show that their approach of analysing behaviour is able to detect tracking that would be missed by EasyList, EasyPrivacy and Disconnect lists.

While they mention GDPR, they did not assess or discuss invisible pixels or the tracking behaviour they observed in its context. We, on the other hand, do not analyse tracking based on behaviour but based on purpose declarations in the context of GDPR compliance. Another difference is that they only investigated third party pixels, while we also consider first party pixels. They did not use machine learning to classify the tracking behaviour but a set of detection methods, while we train a classifier to predict the purpose categories of the tracking pixels. They only investigated pixels of size 1x1, which we showed does not include all tracking pixels. Even though the approach of detecting tracking is different between their study and ours, we both find tracking that would be missed by privacy filter lists.

Matte et al. [20] studied how consent is stored behind the user interface of cookie banners. They specifically focused on banners from CMPs, who

respect IAB Europe's Transparency and Consent Framework (TCF). They assessed the way cookie banners are implemented and store consent in view of potential violations of GDPR or ePrivacy compliance. To do so, they crawled 28'257 European websites for presence of TCF banners, finding 1426 websites with such a banner. These were further searched for potential violations, such as storing positive consent before the user has made their choice, or even registering positive consent when the user has actively opted out. The authors showed that there is at least one potential violation in 54% of the 560 tested websites.

Their study is focused on cookie compliance and they only mention tracking pixels as a way to send consent to third parties. However, they use a range of CMPs, many of which do not specify in their consent notice whether a tracker is a pixel (see table 3.1 for an assessment of whether a CMP specifically declares pixels). Their finding of non-compliance on how consent is stored is, however, likely transferable and the CMP we use in our study, Cookiebot, is one of the CMPs which respects the TCF[1]. We do only look at potential violations in view of wrongly declared purpose, or lack of declarations. Unlike in the study by Matte et al., we do not investigate what the website actually does with the consent. In our webcrawl we always accept all trackers using the Consent-O-Matic extension. We can not distinguish whether the tracker was loaded as a result of our consent or would have been present anyway. It is therefore likely that we only detect a fraction of the violations Matte et al.'s approach would.

Iqbal et al. [16] presented a graph-based machine learning approach to detect and block advertising and tracking resources on the web called AdGraph. They first built a graph representation of the past and present HTML structure, JavaScript behaviour and interrelationship, and previous network requests for each webpage. This was then used to train a classifier to identify advertising and tracking resources. They state that their novel approach has the advantage that it cannot readily be surpassed by countermeasures by adversaries, which might hamper current techniques based on recognising URL patterns or domains. Such a countermeasure is to quickly and frequently change the domain and therefore avoid detection by filter lists.

AdGraph is designed both for online and offline use. An online use could be its deployment in-browser during page execution. An offline use would be to construct filter lists. This means that AdGraph could both be used directly by users to block tracking and advertising resources or it could benefit the user of filter list based adblockers indirectly in that it improves the filter list. They evaluated AdGraph on the Alexa top-10K websites. AdGraph was able to replicate the labels of human-generated filter lists with 95.3% accuracy and even found some mistakes in the filter lists. This is in agreement with

[1] https://www.cookiebot.com/de/iab-framework/

our work, which also found tracking pixels that were missed by filter lists on the one hand and wrongly blocked content on the other.

This study is similar to our work, in that it uses machine learning to be able to detect tracking and advertising in a more robust and possibly more general way than using only patterns from the URL, without needing humans to manually identify these resources. It goes much further in other aspects it uses, such as JavaScript behaviour, than our study. They did this to address adversaries in the real world, which try to evade being detected. While this is beyond doubt very valuable in their case, it is not so important in ours. We do not employ our model in the real world to block tracking but use it to understand compliance with GDPR, unlike Iqbal et al. An adversary tracker as described by Iqbal et al. would make it awkward for websites using those trackers to stay compliant, as they might have to update their consent declarations frequently. Our model on the other hand, might still be able to detect the tracking pixel, as it is almost impossible to change an important feature for the model, such as high entropy, and still be able to track a user. Our model achieves a higher accuracy than Iqbal et al.'s due to specialisation on just one kind of tracker, invisible pixels.

Siby et al. [25] also address the issue of adversarial evasion like the study by Iqbal et al. [16] and directly compare their WebGraph to AdGraph. WebGraph is also graph-based but detects ads and trackers from their action, rather then content. Actions that are necessary for tracking to happen are, for example, storing a unique identifier in the browser or sharing it with another tracker. They showed that WebGraph is much better than AdGraph in detecting tracking and adverts even from adversaries trying actively to evade detection. Overall WebGraph and AdGraph achieve comparable accuracy in recognising ads and trackers.

As mentioned above we do not address adversarial evasion, but we also do not aim to use our model in the real world to detect and block trackers, but instead use it to understand compliance with GDPR. Unlike Siby et al. we only consider content and not actions of trackers.

Chapter 7

Conclusion

We assembled a data set containing tracking pixels found in the wild and their corresponding category label by first checking 1'465'867 domains for presence of CMP Cookiebot. From the 10'380 domains that contained Cookiebot we then extracted purpose labels from consent notices and tracking pixels as well as normal images from HTTP requests and responses. The consent notices and pixels were then matched, and unmatched images were added to the purpose category "necessary".

The resulting 813'162 samples were used to train an XGBoost classifier which achieved high average and mean balanced accuracy (98.77 $\pm$ 0.03% and 95.84 $\pm$ 4.20%, respectively) in distinguishing between the four purpose categories. The resulting model helped to detect important features to recognise tracking pixels by. Among them are a high amount of entropy or information often in form of pseudo-random strings that could serve as unique identifiers to track a user. Another important feature is size and particularly whether the image is 1 pixel large, however, we did also find larger, e.g. 2x2 tracking pixels. Together with other lines of evidence we incorporate these and other features into a set of heuristics on how to recognise tracking pixels in the wild.

We further compared the assessment of privacy threat by the filter lists EasyList and EasyPrivacy, and through the purpose declared in consent declarations. This revealed three main findings. First, and most importantly, they agree to a large extent. Second, taking both the decisions made by filter lists and a majority vote on purpose declared by the consent notices into account can help to confidently detect tracking pixels. This could be used to assign categories to samples that could not be matched to a purpose declaration, or to detect potential violation of GDPR compliance. Third, our approach might be able to better detect some rare tracking pixels. Thus, using our data collection from consent notices might help privacy list maintainers to find rare pixels and keep their lists up to date.

We assessed whether "functional" or "necessary" are meaningful categories for pixels and found that truly "functional" or "necessary" pixels are very rare, but do exist. We cannot, however, demonstrate beyond doubt that the potential candidates for truly "functional" or "necessary" pixels we found should rightfully be in those categories.

Finally, we showed with examples how the collected dataset and models can be used to collect evidence for potential violations of GDPR compliance. We found that between 1.5 and 12% of samples that where included in the "necessary" category could be tracking pixels used for analytics or adverting purposes. We found pixels matched to a potentially wrong outlier label, i.e., deviating from the majority of labels declared by other domains, on 618 different webpages (5.9% of pages visited). More specifically, we found wrong "necessary" purpose labels for the well known Google Analytics pixels on 74 websites (0.7% of pages visited). Of the 10'380 domains we analysed, 6519 or 62.8% contained potential tracking pixels (i.e., images of size 1x1) that could not be matched to a consent declaration and might not be declared.

Building on this study, future work could adapt the classifier to only contain information that is readily available in a browser. Specifically, all of the important features measuring entropy of URL or headers could still be used in a version of our model that is deployed online as part of a browser extension, such as CookieBlock [2], or in an adblocker. An offline use of the model as it currently stands could be to improve existing, or construct new, filter lists. To study further the concerning aspects of potential GDPR violations, it would be enlightening to explore how consent is stored behind the user interface of the consent banner and whether the domains investigated respect the users choices, as done by Matte et al. [20].

Bibliography

[1] Nataliia Bielova, Arnaud Legout, Natasa Sarafijanovic-Djukic, et al. Missed by filter lists: Detecting unknown third-party trackers with invisible pixels. *Proceedings on Privacy Enhancing Technologies*, 2020(2):499–518, 2020.

[2] Dino Bollinger. Analyzing cookies compliance with the GDPR. Master's thesis, ETH Zurich, 2021. https://doi.org/10.3929/ethz-b-000477333.

[3] Dino Bollinger, Karel Kubicek, Carlos Cotrini, and David Basin. Artifacts for "Automating Cookie Consent and GDPR Violation Detection", October 2021. doi:10.5281/zenodo.5568491.

[4] Dino Bollinger, Karel Kubicek, Carlos Cotrini, and David Basin. Automating cookie consent and GDPR violation detection. In *31st USENIX Security Symposium (USENIX Security 22)*, Boston, MA, 2022. USENIX Association. URL: https://www.usenix.org/conference/usenixsecurity22/presentation/bollinger.

[5] BuiltWith. Pivacy compliance usage distribution in the top 1 million sites. URL: https://web.archive.org/web/20201021075918/https://trends.builtwith.com/widgets/privacy-compliance/.

[6] Tianqi Chen and Carlos Guestrin. XGBoost: A scalable tree boosting system. *CoRR*, abs/1603.02754, 2016. arXiv:1603.02754.

[7] OpenWPM contributors. OpenWPM - a webprivacy measurment framework. URL: https://github.com/mozilla/OpenWPM.

[8] Steven Englehardt and Arvind Narayanan. Online tracking: A 1-million-site measurement and analysis. In *Proceedings of ACM CCS 2016*, 2016.

[9] Steven Englehardt and Sandra Siby. Blocklistparser. URL: https://github.com/englehardt/abp-blocklist-parser.

[10] Council of the European Union European Parliament. Directive 2002/58/ec of the european parliament and of the council of 12 july 2002 concerning the processing of personal data and the protection of privacy in the electronic communications sector (directive on privacy and electronic communications). URL: https://eur-lex.europa.eu/eli/dir/2002/58/oj.

[11] European Parliament, Council of the European Union. Regulation (EU) 2016/679 Of the European Parliament and of the Council of 27 April 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing Directive 95/46/EC (General Data Protection Regulation), April 2016. http://data.europa.eu/eli/reg/2016/679/2016-05-04; Last accessed on: 24.01.2022.

[12] fanboy, MonztA, Famlam, and Khrin. Easylist. URL: https://easylist.to/.

[13] Imane Fouad, Nalaliia Bielova, Arnaud Legout, and Natasa Sarafijanovic-Djukic. Tracking the pixels: Detecting web trackers via analyzing invisible pixels. *arXiv preprint arXiv:1812.01514*, 2018. URL: https://hal.inria.fr/hal-01943496v1.

[14] Sanjay Ghemawat and Jeff Dean. LevelDB. URL: https://github.com/google/leveldb.

[15] Jonathan Heawood. Pseudo-public political speech: Democratic implications of the cambridge analytica scandal. *Information Polity*, 23(4):429–434, 2018.

[16] Umar Iqbal, Peter Snyder, Shitong Zhu, Benjamin Livshits, Zhiyun Qian, and Zubair Shafiq. Adgraph: A graph-based approach to ad and tracker blocking. In *2020 IEEE Symposium on Security and Privacy (SP)*, pages 763–776. IEEE, 2020.

[17] Judgement of the Court (Grand Chamber). Case C-673/17 Planet49 GmbH v Bundesverband der Verbraucherzentralen und Verbraucherverbände – Verbraucherzentrale Bundesverband e.V. ECLI:EU:C:2019:246, 1 October 2019. http://curia.europa.eu/juris/document/document.jsf?docid=218462&doclang=EN; Last accessed on: 2021.02.06.

[18] Rolf Bagge Janus Bager Kristensen. Consent-O-Matic. URL: https://github.com/cavi-au/Consent-O-Matic.

[19] Victor Le Pochat, Tom Van Goethem, Samaneh Tajalizadehkhoob, Maciej Korczyński, and Wouter Joosen. Tranco: A research-oriented top sites ranking hardened against manipulation. In *Proceedings of the 26th Annual Network and Distributed System Security Symposium*, NDSS 2019, February 2019. doi:10.14722/ndss.2019.23386.

[20] Célestin Matte, Nataliia Bielova, and Cristiana Santos. Do cookie banners respect my choice?: Measuring legal compliance of banners from iab europe's transparency and consent framework. In *2020 IEEE Symposium on Security and Privacy (SP)*, pages 791–809. IEEE, 2020.

[21] International Chamber of Commerce UK. Icc uk cookie guide. Technical report, International Chamber of Commerce UK, 2012. URL: https://www.cookielaw.org/wp-content/uploads/2019/12/icc_uk_cookiesguide_revnov.pdf.

[22] OneTrust. Cookiepedia. URL: https://cookiepedia.co.uk/.

[23] Adblock Plus. How to write filters. URL: https://help.eyeo.com/en/adblockplus/how-to-write-filters.

[24] Jukka Ruohonen and Ville Leppänen. Invisible pixels are dead, long live invisible pixels! In *Proceedings of the 2018 Workshop on Privacy in the Electronic Society*, pages 28–32, 2018.

[25] Sandra Siby, Umar Iqbal, Steven Englehardt, Zubair Shafiq, and Carmela Troncoso. Webgraph: Capturing advertising and tracking information flows for robust blocking. *arXiv preprint arXiv:2107.11309*, 2021.

[26] Rob van Eijk, Hadi Asghari, Philipp Winter, and Arvind Narayanan. The impact of user location on cookie notices (inside and outside of the European Union). *Workshop on Technology and Consumer Protection (ConPro '19)*, 2019. URL: https://ssrn.com/abstract=3361360.

[27] Ahsan Zafar, Aafaq Sabir, Dilawer Ahmed, and Anupam Das. Understanding the privacy implications of adblock plus's acceptable ads. In *Proceedings of the 2021 ACM Asia Conference on Computer and Communications Security*, pages 644–657, 2021.

[28] Zendesk. Pixel integartion. URL: https://affise.zendesk.com/hc/en-us/articles/115004446425-Pixel-Integartion.

Appendix A

Appendix

A.1 Repositories and Dataset

For the data collection we used the web crawler from Dino Bollinger [4], which can be found in the following GitHub repository. It contains both the presence and consent crawler. A README file explains the content of the repository.

https://github.com/dibollinger/CookieBlock-Consent-Crawler

The feature extraction scripts and classifiers can be found in the following repository. A README file explains the content of the repository.

https://github.com/ganzri/Tracking-Pixels

The dataset collected by the webcrawler containing all the consent label data and observed pixels in a SQL or LevelDB database, respectively, can be found here.

https://zenodo.org/record/6142713

This includes the domains targeted in the webcrawls, the training data in JSON format, and trained classifier models. The README in the folder gives more information.

A.2 Moved Pixels

As discussed in section 3.4.1 we moved samples from a few very common and well known trackers out of the "necessary" and "functional" class to their appropriate class. We determined if and where to move them based on the consent declarations we found on other websites for the respective pixel, some online research, and whether the pixel is blocked by EasyPrivacy.

Pixels from the following netlocs were moved to "analytics".

1. `google-analytics.com` and `ssl.google-analytics.com`
2. `stats.g.doubleclick.net`

Pixels from the following netlocs were moved to "advertising".

1. `facebook.com`
2. `cm.g.doubleclick.net`
3. `track.hubspot.com`: all 572 consent declarations we collected for the domain `hubspot.com` and pixel name `__ptq.gif` give as purpose "advertising".
4. `pagead2.googlesyndication.com`
5. `google.com` and `google.nl`: if the path contained `pagead/1p-conversion`, `ad/ga-audiences` or `pagead/1p-user-list`

A.3 Hyper-parameters used

For reproducible the hyper-parameters used for training model 1 and 2 are given in table A.1

Table A.1: Hyper-parameters used for training with XGBoost

Parameter name	Value
Booster type	'gbtree'
Tree method	'hist'
Learning objective	'multi:softprob'
Evaluation metrics	'merror' and 'mlogloss'
Learning rate	0.3
Maximum tree depth	7
Maximum child weight	3
Maximum delta step	0
Subsample ratio	1.0
Alpha (L1 regularizer)	1
Lambda (L2 regularizer)	0
Tree-growth policy	'depth-wise'
Maximum bins	256

A.4 Comparing Filter Lists and Purpose Declarations

Given are the number of samples per category that are behind figure 5.1 and figure 5.2.

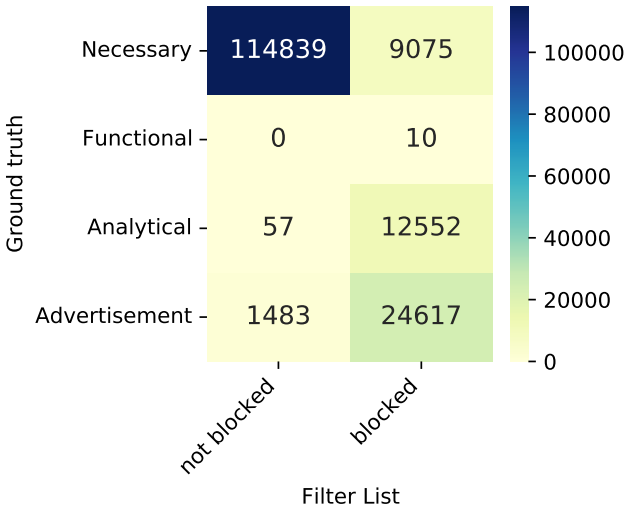

Figure A.1: Comparing consent declarations and filter list decisions. Rows represent the purpose assigned by consent notices (ground truth), columns the decision of filter lists to block or not. The values in row i are the number of samples from purpose category i that are blocked or not by the filter lists EasyPrivacy and EasyList.

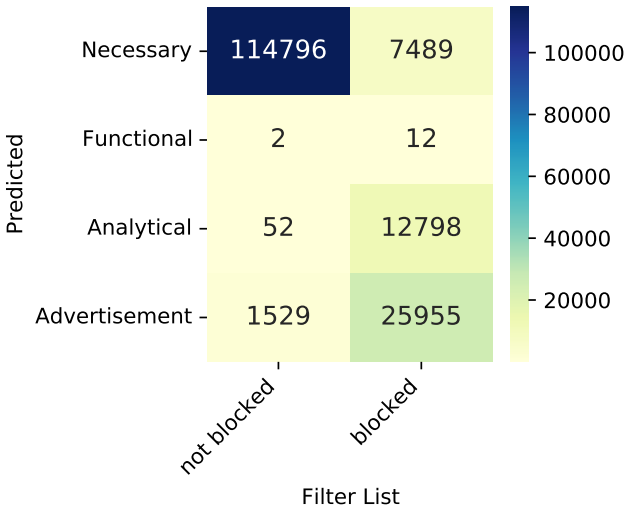

Figure A.2: Comparing model predictions and filter list decisions. Rows represent the purpose category predicted by model 1, columns the decision of filter lists to block or not. The values in row i are the number of samples from predicted purpose category i that are blocked or not by the filter lists EasyPrivacy and EasyList.

Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Declaration of originality

The signed declaration of originality is a component of every semester paper, Bachelor's thesis, Master's thesis and any other degree paper undertaken during the course of studies, including the respective electronic versions.

Lecturers may also require a declaration of originality for other written papers compiled for their courses.

I hereby confirm that I am the sole author of the written work here enclosed and that I have compiled it in my own words. Parts excepted are corrections of form and content by the supervisor.

Title of work (in block letters):

Understanding GDPR compliance of tracking pixel declarations using privacy filter lists

Authored by (in block letters):

For papers written by groups the names of all authors are required.

Name(s):	**First name(s):**
Ganz	Rita

With my signature I confirm that

- I have committed none of the forms of plagiarism described in the 'Citation etiquette' information sheet.
- I have documented all methods, data and processes truthfully.
- I have not manipulated any data.
- I have mentioned all persons who were significant facilitators of the work.

I am aware that the work may be screened electronically for plagiarism.

Place, date	**Signature(s)**
Baar, 18.02.2022	R. Ganz

For papers written by groups the names of all authors are required. Their signatures collectively guarantee the entire content of the written paper.