International Journal of Computer Information Systems and Industrial Management Applications
ISSN 2150-7988 Volume 17 (2025) pp. 14–25

Article

A Comprehensive Analysis for Dark Pattern Detection Using Structural, Visual and Textual Information

Anu Bajaj [1,*], Krish Uppal [1], Rheanca Razdan [1], Yessica Tuteja [1], Ankit Bhardwaj [1] and Ajith Abraham [2]

[1] Computer Science and Engineering Department, Thapar Institute of Engineering and Technology, Patiala 147004, Punjab, India; Krishuppal4@gmail.com (K.U.); Rheancarazdan@gmail.com (R.R.); yessicatuteja@gmail.com (Y.T.); ankitbhardwaj1743@gmail.com (A.B.)
[2] School of AI, Bennett University, Greater Noida 201310, Uttar Pradesh, India; ajith.abraham@ieee.org
* Correspondence author: er.anubajaj@gmail.com

Received date: 11 January 2024; Accepted date: 17 March 2024; Published online: 6 January 2025

Abstract: Dark patterns are deceitful design strategies that control and influence user behavior and have become widespread in digital interfaces across various domains. In this research paper, dark patterns have been classified into broad types: structural dark patterns and UI-based (textual and visual) dark patterns. We present three parallel approaches for detecting the structural and vision based dark patterns effectively. DOM inspector is used for detecting and classifying structural-based dark patterns. For detecting visual dark patterns, YOLOv5 is applied. And finally, for detecting and classifying text-based dark patterns, an EasyOCR and DistilBERT-based approach is presented. We study various websites consisting of different subjects of interest, including e-commerce, news, sports, and business, to analyze the prevalence and types of dark patterns employed. The results and outcomes help us shine a light on the prevalent nature of dark patterns in digital media and provide insights into detecting and reducing their impact on user experience, which will in turn help build trust.

Keywords: dark patterns; e-commerce; dark pattern detection; DOM inspection; DistilBERT; YOLOv5; EasyOCR

1. Introduction

The term ‘Dark Patterns’ was coined by Harry Brignull in 2010. It refers to a user interface that has been crafted meticulously to manipulate users into doing specific tasks that the user might not have done otherwise [1]. In other words, these are deceptive design patterns in the user interface which trick the user to do tasks they might not intend to do. Companies may use dark patterns in their websites to manipulate users into agreeing to certain terms and conditions, which benefit the company. For example, playing videos automatically, making canceling subscriptions difficult, forcing users to create an account on their website, and making cancellation buttons difficult to see are all dark patterns [2]. The purpose behind employing dark patterns can differ, but usually, they revolve around achieving certain goals or outcomes that profit the entity administering them. Some common reasons include:

1. Boosting profits: Dark patterns assist in increasing profits by persuading users into making purchases or choosing expensive add-ons, even if they are unnecessary.
2. Manipulating user behavior: Companies may employ dark patterns to manipulate user behavior for increasing engagement, obtain personal information, or gain consent for data collection.
3. Reducing Transparency: Dark patterns are used to downplay or minimize transparency with respect to service terms, pricing structure, or data handling practices, hence obscuring the user’s

ability to make informed decisions.

4. Gaining Competitive Advantage: Businesses may use dark patterns to obtain a competitive edge by swaying user decisions in favor of their services over those of their competitors.
5. Meeting Performance Metrics: In digital domains like marketing and e-commerce, there is pressure of meeting performance targets such as conversion rates or click-through rates. Dark patterns may be used to inflate these metrics, even at the expense of the trust of users.

In essence, the purpose and reasons behind employing dark patterns involve prioritizing short-term gains or immediate objectives over the long-term interests and well-being of users. Dark patterns threaten the online experience of users by exposing them to increased risks of financial exploitation and misuse of their data by big companies [3]. By making use of these deceitful practices, digital platforms take away the users' right to comprehensive information regarding the services they use. Dark patterns confuse the users, pressure them into subscribing to undesired products or services, act as hindrance in online interactions, drag out simple tasks, and force users to spend more money or reveal more personal information than originally intended [4].

Google, Facebook, Amazon, and LinkedIn are frequently cited as the companies with the highest number of complaints in terms of dark patterns [5]. Companies employing deceptive patterns frequently become entangled in legal proceedings, resulting in substantial fines and penalties. Epic Games settled charges, paying $245 million, related to their alleged use of deceptive patterns within Fortnite's payment system. Noom, a diet app, settled charges by paying $62 million, stemming from allegations of employing deceptive patterns in their subscription and auto-renewal practices. To resolve allegations of adding unauthorized fees for services onto customers' phone bills without their knowledge or consent, AT&T paid $105 million in settlement charges [6]. It is therefore crucial to combat dark patterns for protecting user rights and promoting transparency. Therefore, in this paper, a method is proposed to detect dark patterns on websites, which deals with nine patterns, categorized into two domains: structural and textual as shown in Figure 1.

1. Pop-ups or Overlays: Pop-ups become dark patterns when they're used deceptively or manipulatively to compel user actions or disrupt their browsing experience. This undermines user trust and diminishes the usability of websites or applications.
2. Hidden Elements: This Dark Pattern obscures user-relevant options or actions, disguising them as inconsequential details. It may involve hiding relevant information within fine print, terms and conditions, or subtly altering text color to diminish visibility.
3. Misleading Labels or Buttons: Some websites employ deceptive button designs, which appear to direct users to one page but instead lead them to a different destination, such as a sign-up page.
4. Pre-selected Checkboxes: The user is presented with a checkbox that has already been selected for them, in order to influence their decision-making.
5. Misdirection: This deceptive design tactic manipulates your focus towards one element, potentially causing you to overlook other important aspects.
6. Obstruction: The user encounters obstacles that hinder their ability to accomplish their task or obtain the information they seek.
7. Fake Scarcity: The user feels compelled to take action due to encountering a deceptive representation of limited supply or popularity, despite its falsity.
8. Fake social proof: The user is deceived into perceiving a product as more popular or trustworthy than its actual standing by encountering fabricated reviews, testimonials, or activity notifications.
9. Fake urgency: The user feels compelled to complete an action due to the presentation of an artificial time constraint, despite it being fabricated.

Dark Patterns Classification	
Structural Dark Patterns	**Textual Based Dark Pattern**
1. Pop-Ups or Overlays 2. Hidden Elements 3. Misleading Labels or Buttons 4. Pre-Selected Checkboxes	1. Misdirection 2. Obstruction 3. Fake Scarcity 4. Fake Social Proof 5. Fake Urgency

Figure 1. Dark Patterns Classification.

The main contribution towards the detection of dark pattern includes the integration of three parallel approaches, showcasing a comprehensive integration of Document Object Model Inspection, Natural Language Processing and Computer Vision:

- To detect and classify structural based dark patterns like Pop-ups or Overlays, Hidden Elements, Misleading Labels or Buttons, and Pre-selected Checkboxes, using the proposed Document Object Model (DOM) Inspector.
- To identify visual dark patterns usingYOLOv5 (You Only Look Once version-5).
- To detect and classify text based dark patterns like misdirection, obstruction, fake scarcity, fake social proof, and fake urgency by employing Easy Optical Character Recognition (EasyOCR) and DistilBERT.

The rest of the paper is organized as follows. In Section 2, the related works in the field of dark patterns is presented. Section 3 describes the experimental setup. We detail the methodology used in Section 4. The proposed work for dark pattern detection is described in Section 5. The experimental results and their analysis is shown in Section 6. Finally, the conclusion of the paper in Section 7.

2. Related Work

The related work in dark patterns can be summarized in three categories: dark pattern taxonomies, detecting dark patterns using deep learning and automated detection in mobile UIs as presented below:

2.1. Dark Pattern Taxonomies

Mathur et al. [7] used webcrawler to visit 11,000 most popular shopping websites, found and analyzed dark patterns manually and created a large dataset of dark patterns. Their dataset consisted of 1,818 dark pattern instances across 53 thousand pages. They revealed 15 types of dark patterns in 7 broad categories. Also, they documented the third-party entities that enable the dark pattern on these websites. However, the study was specific to websites of one domain and was difficult to generalize. Also, the dataset consisted of only dark pattern entries that were examined manually.

Gunawan et al. [8] also manually investigated dark patterns. However, the study focused on online services available across desktop browsers, mobile browsers and google play store ensuring equivalent functionality across all modalities. The dark patterns were identified based on coercion, steering, deception and strategic omission that favored the service over the user. The features analyzed in the study include assessing feature parity across different modalities, examining dark pattern causality and impact, and investigating privacy implications. The study also explored how design decisions can impact user experiences and behaviors, particularly in terms of autonomy, privacy, and control.

The researchers identified certain challenges in their study which involved the framing of designs that can steer designers towards developing darker designs, the impact of design on user experience, and the complexities of assessing dark patterns that are observed repeatedly. The dataset consisted of

105 services across various categories from the Play Store. These services were analyzed to compare dark patterns across desktop web, mobile web, and mobile app versions. The researchers also developed a codebook of dark patterns during their study to assess their frequency and impact on user behavior. These manual examinations of dark patterns proved to be tiresome, which led to the development of automated systems for dark pattern detection as elucidated below:

2.2. Automated Detection Using Deep Learning

Yuki et al. [9] used classical natural language processing methods: Support Vector Machine(SVM), Random Forest, Gradient Boosting (LightGBM) and Transformer Based Pre-Trained Language Models Bidirectional Encoder Representations from Transformers (BERT), Robustly Optimized BERT Approach (RoBERTa), A Lite BERT (ALBERT) and eXtreme Learning Network (XLNet) to develop an automated system for dark pattern detection. They modified the dataset manually constructed by Mathur et al. [7] and added the collection of negative samples (non-dark patterns) to create a comprehensive textual dataset. According to the study, the RoBERTa-large model achieved the highest accuracy of 0.975.

Yuki et al. [10] further extended their study and combined the transformer-based pretrained models BERT [11] and RoBERTa [12] with post-hoc explanation techniques Local Interpretable Model-Agnostic Explanations (LIME) [13] and SHapley Additive exPlanations (SHAP) [14] to develop a dark pattern auto detection system. Classification layer was added to the BERT output layer. The output of the classification layer was trained to determine whether the input text is a dark pattern or not. They extracted the terms influencing dark patterns using SHAP-based explanation. They further analyzed the calculated explainability and extracted the terms that determined the presence of dark patterns. The extracted terms were identified under four categories: fear of missing out, consensus on popularity, sense of urgency, and special offers [15]. It was observed that SHAP outperformed LIME as it extracted the terms that influence the prediction of text being dark pattern or not.

Mansur et al. [16] used Faster R-CNN and natural language processing techniques to recognize a set of visual and textual cues in application screenshots. They used this model to detect, classify and localize 10 unique UI dark patterns. They developed Aid for detecting UI dark patterns (AIDUI) to detect dark patterns through a fully automated process that operated only on visual data (screenshot). They used a ContextDP dataset of dark pattern instances localized to UI screenshots consisting of both dark patterns and non-dark patterns. The study concluded that text, color and spatial analysis gave the best result [17]. However, the proposed model gave poor performance for mobile application-based UIs with lesser overall average precision, recall and F1-score.

2.3. Automated Detection in Mobile UIs

Chen et al. [18] developed UIGuard, an automated dark pattern detection system that employed computer vision and natural language pattern matching techniques to detect dark patterns. The study focused on mobile UI and gave higher accuracy for detecting dark patterns in mobile UIs. The dataset consisted of 4,999 non-dark pattern UIs and 1,353 malicious UIs of 1,660 instances spanning 1,023 mobile apps. They also aimed to increase users' knowledge of dark patterns through a user study. However, the study was specific to mobile UI. They could not overcome the challenge of different views of users on what constitutes a dark pattern. The summary of the related work is presented in Table 1.

Table 1. Dark Patterns Related works.

Author and Year	Adopted Scheme	Features	Challenges	Dataset
Mathur et al. (2019) [7]	Web crawling by visiting 11K shopping websites	15 types of dark patterns in 7 categories	Specific to one domain, difficult to generalize. Manual examination	Manually created a large dataset with 1,818 dark pattern instances across 53 thousand pages from 11 thousand websites.
J Gunawan et al. (2021) [8]	Manual investigation. Multimodal, coercion, steering, deception and strategic omission	Feature parity across different modalities, dark pattern causality and impact, investigating privacy implications	Design tasks can steer designers towards adopting darker designs, impact of design decisions on users	105 service from the Play Store. Across desktop web, mobile web, and mobile app versions.
Yada et al. (2022) [9]	Machine learning models like BERT, RoBERTa, ALBERT and XLNet.	Non-dark pattern text from e-commerce websites	Non dark pattern data collection, segmentation, feature selection	Negative samples (non-dark patterns) along with manually constructed dataset
Yada et al. (2023) [10]	BERT, RoBERTa LIME and SHAP	Explainability, extracted terms, SHAP outperforms LIME	Non dark pattern data collection, segmentation	Negative samples (non-dark patterns) along with manually constructed dataset of Mathur et al. [7]
Mansur et al. (2023) [16]	Recognizes visual and textual cues in app screenshots. Detects, classifies and localizes UI dark patterns.	AidUI, automated process. Operates on visual data (screenshot)	Poor performance for mobile based UI's. Lesser overall average precision, recall and F1-score.	ContextDP Dataset, UI screenshots, non-dark pattern instances included.
Chen et al. (2023) [18]	UIGuard, Automated	Mobile UI. Higher accuracy.	Specific to mobile UI, Different views of users, Potential misuse	4,999 non- dark pattern UIs and 1,353 malicious UIs of 1,660 instances spanning 1,023 mobile apps.

It was observed from the Table 1 that different researchers worked on different types of dark patterns, while these works have their own pros and cons associated with their methodology, we tried to grasp the fundamental essence of each approach and narrow it down to make a more efficient, robust, and faster dark-pattern detection system for real-time application. We also curated a comprehensive self-made dataset that includes non-dark pattern instances along with dark pattern instances. An automated detection system is built using easyOCR, DistilBERT for textual analysis, and YOLOv5 for visual analysis, along with diving into structural analysis of the webpage using a DOM Inspection strategy for

detecting more types of dark patterns. In other words, the multimodal approach, not limiting to textual or visual detection using deep learning, but also extending to DOM inspection techniques proved largely effective in detecting a multitude of types of Dark Patterns.

3. Experimental Setup

The experiment was performed on a Microsoft Windows 11-based system with AMD Ryzen 5 Hexa Core R5-5600H Processor and 8GB RAM. The deep learning models were trained and deployed on Jupyter Notebook using Python 3.8. The DOM inspection program was made on VisualStudios in JavaScript Language. Two types of datasets have been used for experiment as described below:

Dataset 1 (Image dataset): It encompasses 490 self-annotated images, providing a comprehensive collection of various web pages exhibiting dark patterns. These images are a valuable resource for training and evaluating the dark pattern detection system. The dataset was further split into three parts: training set, validation set, and test set. The training set has 390 images, the validation set has 40 images, and the test set has 60 images, with their respective annotations.

Dataset 2 (Dark Patterns text dataset): We used the dataset constructed by Yuki et al. [9]. It offers a text dataset that adds depth and diversity to the training process. This textual information enhances the system's capability to recognize and classify dark patterns embedded in language across different web interfaces. To train this dataset on the proposed models, Some data preprocessing steps need to be performed to make it fit for training as the dataset consisted of unbalanced minority classes:

4. Methodology Used

4.1. You Only Look Once version 5 (YOLOv5)

YOLO revolutionized object detection by reframing it as a unified regression task, directly predicting bounding box coordinates and class probabilities from image pixels. With YOLO, a single glance at an image is all it takes to forecast the presence and positions of objects. Employing a single convolutional network, YOLO concurrently predicts multiple bounding boxes and their associated class

probabilities [19]. YOLO trains on full images and directly optimizes detection performance.

According to Redmon et al. [19] YOLO offered numerous advantages compared to conventional object detection techniques. First, YOLO achieved remarkable speed due to its regression-based approach, eliminating the need for intricate pipelines. Additionally, YOLO surpassed other real-time systems by achieving over twice the mean average precision. Second, YOLO adopted a global perspective on the image while making predictions. Unlike sliding window and region proposal-based methods, YOLO processed the entire image during both training and testing phases, thus inherently capturing contextual information about classes and their appearances. In contrast, Fast R-CNN, a leading detection technique [20], often misidentified background patches as objects due to its inability to consider larger contextual cues. Consequently, YOLO committed fewer than half the number of background errors compared to Fast R-CNN. Third, YOLO excelled at learning versatile object representations. Its high generalizability reduced the likelihood of breakdowns when applied to novel domains or

unanticipated inputs.

After 4 subsequent versions, YOLOv5 [21] was developed using Pytorch [22] Ultralytics released it as an open-source project. During testing on the MS COCO dataset [23] test-dev 2017, YOLOv5x achieved a notable average precision (AP) of 50.7% using an image size of 640 pixels, making it one of the most successful object detection deep learning models ever. Unlike its predecessors, YOLOv5 adopts a lightweight and efficient architecture. It uses a combination of CSPDarknet53 [24] as the backbone network and a general YOLO head from its predecessors for object detection. Its architecture, training techniques, and large training dataset made it much more accurate and precise than its preceding YOLO versions. It is specifically designed to be faster and more efficient, making it ideal for real-time applications. So, in this study, it is chosen for dark-pattern detection in UI. The YOLOv5 architecture is shown in Figure 2.

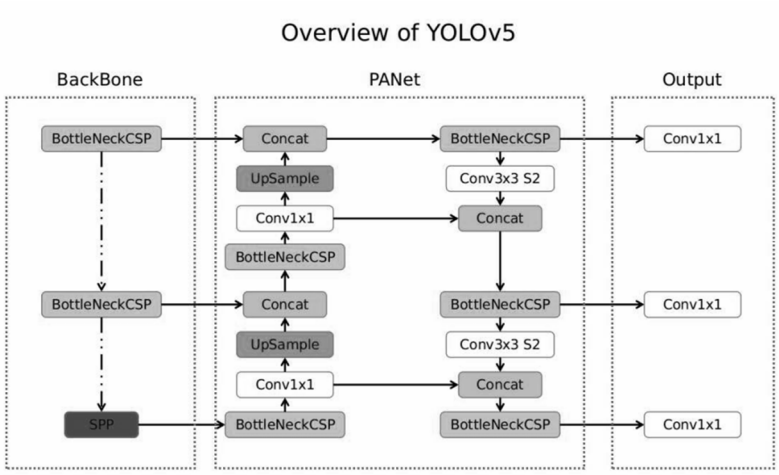

Figure 2. YOLOv5 Architecture.

4.2. DistilBERT

Jacob et al. [25] developed Bidirectional Encoder Representation from Transformers (BERT) by stacking the encoders from the transformer architecture together . BERT was a significant advancement in the field of natural language processing. Unlike traditional unidirectional models, BERT considered both left and right context in all layers, allowing it to capture a more comprehensive understanding of words in a sentence in less time. The BERT is trained in two phases: Pre-training to understand language and Fine-tuning to learn specific tasks. BERT is pre-trained on a large text corpus using a masked language model (MLM). After pre-training, BERT is fine-tuned for specific tasks by adding task-specific input and output layers by adjusting the model's parameters to adapt to the nuances of the target task.

Later, Victor et al. [26] introduced DistilBERT, a lighter and faster version of BERT. DistilBERT can reduce the size of the BERT model by 40% while being 60% faster and preserving nearly 97% of BERT's language learning abilities. The key idea behind DistilBERT is to reduce the size and computational resources required for inference; that is, it is more feasible to deploy in resource-constrained environments such as the one discussed in this paper, i.e., making it highly ideal for textual dark pattern detection. The DistilBERT model was imported using Google's BERT library, and was trained on our custom Dark Patterns dataset. Thus, due to its better speed and lighter model along with its efficient natural language understanding ability, DistilBERT proves to be a good fit for developing automated dark pattern detection systems.

4.3. EasyOCR

EasyOCR is a Python library-based Optical Character Recognition (OCR) model that detects text of many languages from images [27]. OCR at its core, is a deep learning-based technology that enables the extraction of text content from different types of media, such as web images, and screenshots, for analysis and further processing. EasyOCR aims to simplify the process of integrating OCR capabilities into Python applications by offering a user-friendly API.

The ease of use, high accuracy, and speed of the EasyOCR model made it highly suitable for text extraction from web images. The text extracted from web images will be sent to the custom-trained DistilBERT model for detection of textual dark patterns.

5. Proposed Work

The proposed work incorporates three parallel approaches. First, we employ the DOM Inspector (DOMi) to inspect the Document Object Model (DOM) tree of HTML and XML-based documents to detect anomalies that indicate deceptive design elements. Secondly, for detecting textual based dark patterns, we use EasyOCR, for extracting text from web screenshots, and DistilBERT, for further processing the text and categorizing dark patterns into various classes. Thirdly, we use YOLOv5, for detecting visual based dark patterns from the web screenshot. The combined result from these approaches results in a comprehensive dark pattern detection as shown in Figure 3.

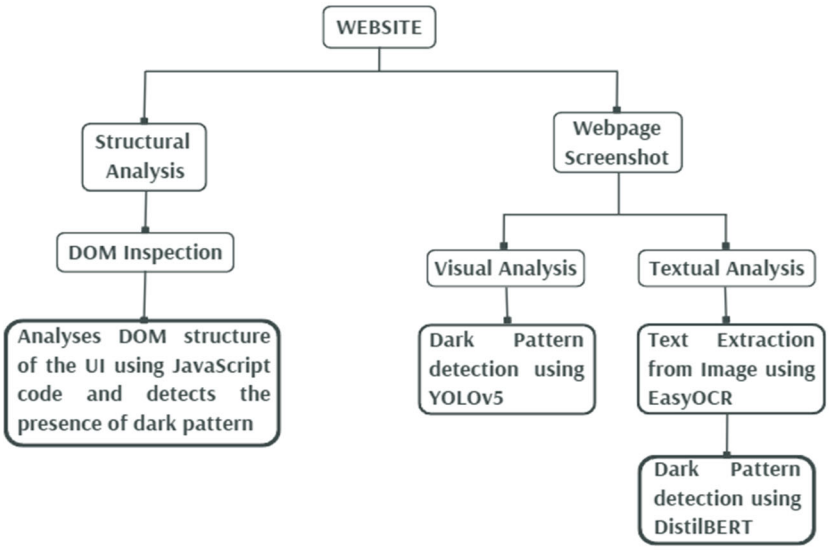

Figure 3. Flowchart of Approach.

5.1. Structural Patterns: DOM Inspector

The Javascript code is written for detecting dark patterns as shown in Figure 4. It goes through the DOM files of a website and looks for various instances/sections embedded in the structure of the web page's code that are usually made/included to create dark patterns that intentionally deceive users into selecting options/instructions as per the host's choice. To ensure that the script runs after the HTML content has been fully loaded, 'Function Execution on DOM Ready' is called when the DOM is ready (DOMContentLoaded event).

```
// Function to identify potential dark patterns
function identifyDarkPatterns() {
  // Check for pop-ups or overlays
  const popups = document.querySelectorAll('.popup, .overlay');
  if (popups.length > 0) {
    console.warn('Possible pop-up or overlay:', popups);
  }

  // Check for hidden elements
  const hiddenElements = document.querySelectorAll('[style*="display:none" i],
[style*="visibility:hidden" i]');
  if (hiddenElements.length > 0) {
    console.warn('Hidden elements found:', hiddenElements);
  }

  // Check for misleading labels or buttons
  const misleadingElements = document.querySelectorAll('[data-deceptive="true"]');
  if (misleadingElements.length > 0) {
    console.warn('Misleading elements found:', misleadingElements);
  }

  // Check for forced actions (e.g., pre-selected checkboxes)
  const preSelectedCheckboxes = document.querySelectorAll('input[type="checkbox"]:checked');
  if (preSelectedCheckboxes.length > 0) {
    console.warn('Pre-selected checkboxes found:', preSelectedCheckboxes);
  }

  // Log a message if no dark patterns are found
  if (
    popups.length === 0 &&
    hiddenElements.length === 0 &&
    misleadingElements.length === 0 &&
    preSelectedCheckboxes.length === 0
  ) {
    console.log('No dark patterns detected.');
  }
}

// Call the function to identify dark patterns
identifyDarkPatterns();
```

Figure 4. DOM Inspection code.

The Javascript code detects four different types of dark patterns, namely:

1. Pop-ups or Overlays: It starts by querying the DOM for elements with the class ‘popup’ or ‘overlay.’
2. Hidden Elements: It looks for elements with styles containing ‘display: none’ or ‘visibility: hidden.’
3. Misleading Labels or Buttons: It searches for elements with the attribute ‘data-deceptive’ set to “true”.
4. Pre-selected Checkboxes: It identifies pre-selected checkboxes by querying for input elements of type ‘checkbox’ that are currently checked.

If any potential dark patterns are found during the checks, the script logs messages to the console, indicating the type of dark pattern and providing details about the identified elements. Otherwise, it logs a message indicating no dark patterns were found. The samples are shown in Figures 5 and 6.

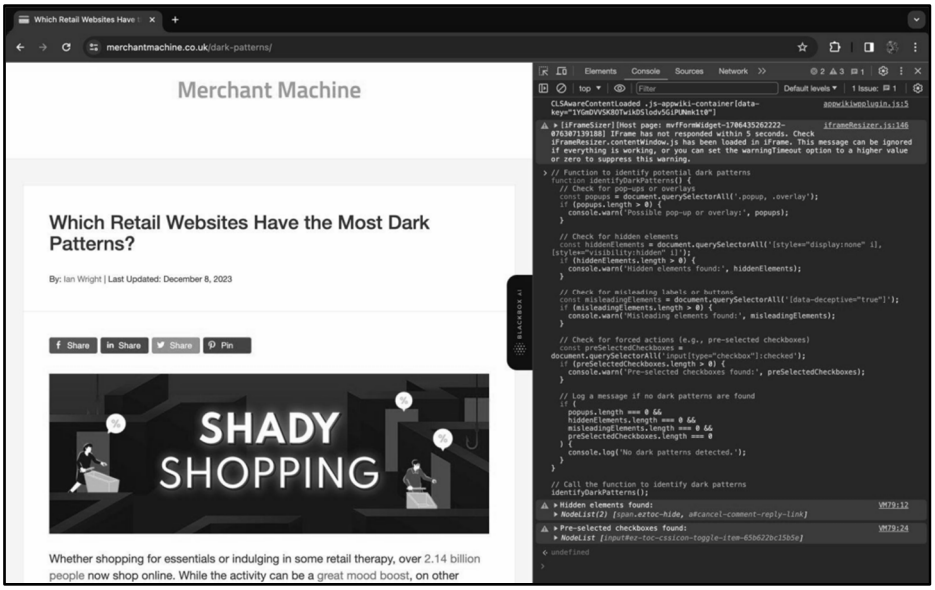

Figure 5. DOM Inspection Sample 1.

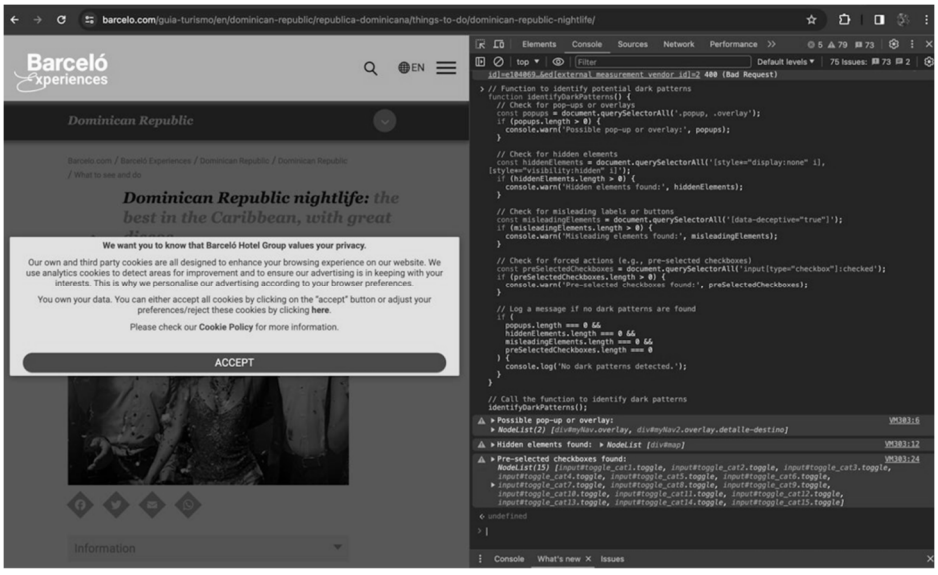

Figure 6. DOM Inspection Sample 2.

5.2. Textual Based Dark Patterns: EasyOCR and DistilBERT

It is divided into two parts: text extraction followed by textual pattern analysis.

Text Extraction: For the task of text extraction we use EasyOCR. Optical Character Recognition (OCR) is the electronic conversion of images of typed, handwritten, or printed text into machine-encoded

text. EasyOCR is an open-source OCR Tool implemented using PyTorch, and it supports 80+ languages. We use this Python OCR module to extract textual content from web pages. This step is crucial for detecting dark patterns that rely on deceptive text elements.

Textual Pattern Analysis: The extracted text is then fed into custom-trained DistilBERT. DistilBERT is a lighter and faster version of BERT (Bidirectional Encoder Representations from Transformers), which provides a contextual understanding of the text. DistilBERT can reduce the size of the BERT model by 40% while being 60% faster, all the while preserving 97% of BERT's language understanding capabilities. Our DistilBERT model is fine-tuned on a custom dataset curated for identifying and classifying different types of dark patterns present in the text. The model analyzes the linguistic structure, semantics, and context of the extracted text to determine the presence and nature of deceptive practices. It classifies the dark patterns into six categories, namely misdirection, obstruction, scarcity, social proof, and urgency. The code is shown in Figure 7.

```
model_path = "BERT_TextModel"
model = DistilBertForSequenceClassification.from_pretrained(model_p
tokenizer = DistilBertTokenizer.from_pretrained(model_path)
input_string = extracted_text
tokenized_input = tokenizer.encode_plus(
    input_string,
    add_special_tokens=True,
    max_length=256,
    padding='max_length',
    return_attention_mask=True,
    return_tensors='pt'
)

model_output = model(**tokenized_input)
logits = model_output.logits
predicted_class = torch.argmax(logits, dim=1).item()
class_mapping = {
    0: 'Misdirection',
    1: 'Not Dark Pattern',
    2: 'Obstruction',
    3: 'Scarcity',',
    4: 'Social Proof',
    5: 'Urgency'
}
predicted_label = class_mapping.get(predicted_class, 'Unknown')
print(f"Dark Pattern: {predicted_label}")
```

```
Dark Pattern: Misdirection
```

Figure 7. Textual dark pattern detection using DistilBERT.

5.3. Visual Based Dark Patterns: YOLOv5

YOLOv5 is the fifth iteration of the "You Only Look Once" object detection model. It is fast, accurate and helps in tasks such as object detection, instance segmentation and image classification. We trained the model on a custom dataset created for identifying visual patterns associated with dark patterns. The dataset includes examples of highlighted boxes, misleading buttons, intrusive cookie pop-ups, fake banners, and other deceptive elements commonly used to manipulate user behavior on websites. YOLOv5 efficiently identifies and highlights these elements as shown in Figures 8 and 9.

Figure 8. Sample 1: Original Image versus Visual Dark Pattern Detection using YOLOv5.

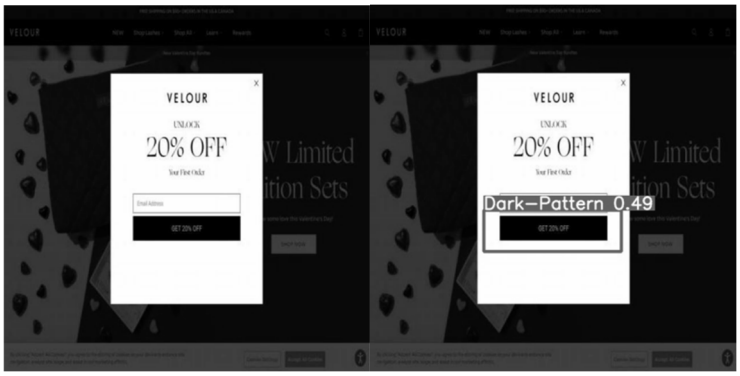

Figure 9. Sample 2: Original Image versus Visual Dark Pattern Detection using YOLOv5.

Results from YOLOv5 and DistilBERT are integrated together with results from DOM Inspector for the final result evaluation of a web page.

6. Data Analysis and Results

For DOM inspection, websites have been classified into 4 broad domains: E-commerce, Sports, Business and News. JavaScript code for identifying dark patterns is run on the console of the chrome browser for each website. The code detects the presence of dark patterns in the webpage and their type. This process is repeated for 20 websites each, for each domain. The values of each kind of dark pattern is normalized in the range of 0 to 1, with 0 meaning no presence of that dark pattern in any of the websites of the respective domain and 1 meaning the dark pattern is detected in all the websites analyzed for the respective domain. The output is recorded in Table 2.

Table 2. Evaluation Metrics for DOM Inspection.

Domain	Hidden Element	Pre-Selected Checkbox	Pop-Ups	No Dark Pattern Found
E-commerce	0.8	0.2	0.1	0.2
Sports	0.7	0	0.1	0.3
Business	0.6	0.1	0.2	0.4
News	1	0	0.2	0

It is observed from Table 2 that the DOM Inspection technique works best for detecting hidden elements in websites across all domains. This technique works well for detecting Pop-Ups as well. But, DOM Inspection doesn't prove to be a suitable technique for detecting Pre-Selected checkboxes'

dark patterns.

DistilBERT was run on the textual dataset constructed by Yuki et al. [9]. The dataset was split into two sets: 80% for training and 20% for testing. DistilBERT performed multiclass classification and classified the dark patterns into 6 categories Misdirection, Obstruction, Scarcity, Social Proof and Urgency along with non-dark pattern instances. The evaluation metrics for model performance on the test set was recorded in Table 3.

Table 3. Evaluation Metrics of DistilBERT over each class.

Dark Pattern	Precision	Recall	F1-Score	Instances
Not Dark Pattern	0.96	0.97	0.96	244
Misdirection	0.77	0.79	0.78	38
Obstruction	1	0.83	0.91	6
Scarcity	0.98	1	0.99	81
Social Proof	1	0.96	0.98	54
Urgency	0.98	0.96	0.97	47
Accuracy			0.95	470
Macro Average	0.81	0.79	0.80	470
Weighted Average	0.95	0.95	0.95	470

It is observed from Table 3 that DistilBERT gives an overall accuracy of 95% for the textual dataset and detects Obstruction, Scarcity, Social Proof and Urgency dark patterns with a higher precision, recall and F1-Score as compared to Misdirection.

Visual Based Detection

We trained the YOLOv5 model on a custom dataset comprising 490 images meticulously annotated by ourselves. The dataset encompasses a variety of instances showcasing highlighted boxes, misleading buttons, intrusive cookie pop-ups, fake banners, and other deceptive elements frequently employed to influence user actions on websites. We divided the dataset into three distinct subsets: training, validation, and test sets. The training set comprises 390 images, the validation set contains 40 images, and the test set consists of 60 images. Each subset is accompanied by annotations tailored to its respective images. YOLOv5 was run on the test set, which then detected the presence of visual based dark patterns and its evaluation metrics was recorded in Table 4.

Table 4. Evaluation Metrics of YOLOv5.

Precision	Recall	mAP 50	mAP 50-95
0.615	0.48	0.542	0.207

It is observed from Table 4 that YOLOv5 gives:

- Moderate precision with 61.5% of detected objects are correct.
- Moderate recall with 48% of actual positive instances in the dataset.
- Average mAP 50 score, indicating 54.2% precision when considering Intersection over Union (IoU) threshold of 0.5.
- Model's performance drops for the current dataset when considering a wider range of IoU thresholds from 0.5 to 0.95 with a mAP 50-95 score of 0.207.

7. Conclusions

In this paper, we categorized dark patterns into two distinct categories: structural and UI Based, which are further divided into textual based and visual based dark patterns. A self-annotated dataset was developed that serves as a valuable resource for training and evaluating the dark pattern detection system. The framework consisted of three parallel approaches for detecting dark patterns: DOM Inspector for structural dark patterns, DistilBERT for textual dark patterns and YOLOv5 for visual dark patterns. The websites were studied for different areas of focus and analyzed the types of dark patterns they employed. The proposed work illustrates a novel approach integrating the results from three methodologies that ensures ethical conduct, enhances user experience, helps businesses in complying with regulations and promotes fair competition in the digital realm. In future, we will try to apply more deep learning algorithms to identify the dark patterns.

Author Contributions

Conceptualization and Methodology: A.B. (Anu Bajaj); Project Administration: A.B. (Anu Bajaj) and Y.T.; Software Development: A.B. (Ankit Bhardwaj) and K.U.; Resources: Y.T. and R.R.; Data Curation: A.B. (Ankit Bhardwaj); Supervision: A.A. All authors have read and agreed to the published version of the manuscript.

Funding

This research received no external funding.

Conflict of Interest Statement

The authors declare no conflict of interest.

Data Availability Statement

The dataset will be available on request.

References

1. I. Chordia, L.P. Tran, T.J. Tayebi, E. Parrish, S. Erete, J. Yip and A. Hiniker, 2023, April. Deceptive design patterns in safety technologies: A case study of the citizen app. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems,* pp. 1-18.
2. I. Karagoel and D. Nathan-Roberts, 2021, September. Dark patterns: Social media, gaming, and e-commerce. In *Proceedings of the Human Factors and Ergonomics Society Annual Meeting,* Vol. 65, No. 1, pp. 752-756.
3. Packetlabs, 2023. What Are The Risks Of "Dark Pattern" Social Engineering Techniques? Available at: https://www.packetlabs.net/posts/dark-pattern-social-engineering (Accessed on 14 April 2024).
4. A. Ramachandran. Dark Pattern and It's Impact On ECommerce. Codilar Technologies. https://www.codilar.com/dark-pattern-and-its-impact-on-ecommerce/ (Accessed on 14 April 2024).
5. C. Calabrese. *GitHub - Carmineh/Dark-Pattern-Identifier: Browser Extension that identifies Dark Pattern*. GitHub. https://github.com/Carmineh/Dark-Pattern-Identifier (Accessed on 14 April 2024).

6. AT&T To Pay $105 Million To Resolve Wireless Cramming Investigation. Federal Communications Commission. October 11, 2018. https://www.fcc.gov/document/att-pay-105-million-resolve-wireless-cramming-investigation-0 (Accessed on 14 April 2024).
7. Mathur, A., Acar, G., Friedman, M.J., Lucherini, E., Mayer, J., Chetty, M. and Narayanan, A., 2019. Dark patterns at scale: Findings from a crawl of 11K shopping websites. *Proceedings of the ACM on Human-Computer Interaction*, 3(CSCW), pp. 1-32.
8. J. Gunawan, A. Pradeep, D. Choffnes, W. Hartzog and C. Wilson, 2021. A comparative study of dark patterns across web and mobile modalities. In *Proceedings of the ACM on Human-Computer Interaction*, 5(CSCW2), pp.1-29.
9. Y. Yada, J. Feng, T. Matsumoto, N. Fukushima, F. Kido and H. Yamana, 2022, December. Dark patterns in e-commerce: a dataset and its baseline evaluations. In *Proceedings of the 2022 IEEE International Conference on Big Data (Big Data),* pp. 3015-3022.
10. Y. Yada, T. Matsumoto, F. Kido and H. Yamana. 2023, December. Why is the User Interface a Dark Pattern?: Explainable Auto-Detection and its Analysis. In *Proceedings of the 2023 IEEE International Conference on Big Data (BigData),* pp. 6308-6310.
11. J. Devlin, M.W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv* 2018, arXiv:1810.04805.
12. Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer and V. Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv* 2019, arXiv:1907.11692.
13. M.T. Ribeiro, S. Singh and C. Guestrin, 2016, August. "Why should i trust you?" Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pp. 1135-1144.
14. S.M. Lundberg and S.I. Lee, 2017. A unified approach to interpreting model predictions. *Advances in neural information processing systems*, 30.
15. W.C. Koh and Y.Z. Seah, 2023. Unintended consumption: The effects of four e-commerce dark patterns. *Cleaner and Responsible Consumption, 11*, p.100145.
16. S.H. Mansur, S. Salma, D. Awofisayo and K. Moran, 2023, May. Aidui: Toward automated recognition of dark patterns in user interfaces. In *Proceedings of the 2023 IEEE/ACM 45th International Conference on Software Engineering (ICSE),* pp. 1958-1970.
17. GitHub. (n.d.). AidUI/README.md at master · SageSELab/AidUI. GitHub. https://github.com/SageSELab/AidUI/blob/master/README.md (Accessed on 14 April 2024).
18. J. Chen, J. Sun, S. Feng, Z. Xing, Q. Lu, X. Xu and C. Chen, 2023, October. Unveiling the Tricks: Automated Detection of Dark Patterns in Mobile Applications. In *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology,* pp. 1-20.
19. J. Redmon, S. Divvala, R. Girshick and A. Farhadi, 2016. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition,* pp. 779-788.
20. S. Ren, K. He, R. Girshick and J. Sun, 2015. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in Neural Information Processing Systems,* 28.
21. G. Jocher, 2020. ultralytics/yolov5. [online] GitHub. 2020. Available at: https://github.com/ultralytics/yolov5 (Accessed on 14 April 2024).
22. A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, and A. Desmaison, 2019. Pytorch: An imperative style, high-performance deep learning library. *Advances in Neural Information Processing Systems,* 32.
23. T.Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár and C.L. Zitnick, 2014. Microsoft coco: Common objects in context. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13,* pp. 740-755.
24. W. Chien-Yao, L. Hong-Yuan Mark, W. Yueh-Hua, C. Ping-Yang, H. Jun-Wei, Y. I-Hau. 2020. CSPNet: A new backbone that can enhance learning capability of CNN. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops,* pp. 390-391.
25. J. Devlin, M.W. Chang, K. Lee and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv* 2018, arXiv:1810.04805.
26. V. Sanh, L. Debut, J. Chaumond and T. Wolf. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. *arXiv* 2019, arXiv:1910.01108.
27. GitHub. (n.d.). GitHub - JaidedAI/EasyOCR: Ready-to-use OCR with 80+ supported languages and all popular writing scripts including Latin, Chinese, Arabic, Devanagari, Cyrillic and etc. [online] Available at: https://github.com/JaidedAI/EasyOCR (Accessed on 14 April 2024).