World Journal of Advanced Engineering Technology and Sciences

eISSN: 2582-8266

Cross Ref DOI: 10.30574/wjaets

Journal homepage: https://wjaets.com/

(Research Article)

Surf Shelter: A Big Data-driven Risk Assessment System Using Multi-label Classification

Aditya Karki [1,*], Ayesha Imam [2,] and Navaraj Pandey [2]

[1] *Math and Computer Science, Fisk University, Nashville, TN 37208.*
[2] *Computer Science, Fisk University, Nashville, TN 37208.*

World Journal of Advanced Engineering Technology and Sciences, 2025, 16(03), 373-385

Publication history: Received on 11 August 2025; revised on 16 September 2025; accepted on 19 September 2025

Article DOI: https://doi.org/10.30574/wjaets.2025.16.3.1349

Abstract

Web content moderation faces increasingly diverse threats ranging from phishing and malware to clickbait and fraud. This project, Surf Shelter, proposes a unified risk assessment system that utilizes multi-label classification to detect multiple types of website threats simultaneously. By leveraging big data from sources such as Common Crawl, OpenPageRank, GitHub, VirusTotal, PhishTank, and Google Safe Browsing, we compile a comprehensive dataset of websites and threat intelligence. We extract rich features using natural language processing (DistilBERT embeddings), static code analysis, and security heuristics, and apply an ensemble soft-voting strategy to label websites across threat categories. Preliminary results on 11,500 collected webpages (with 1,000 labeled) show that our model can achieve high overall accuracy (around 84%) in identifying malicious content, though calibration is needed to reduce false positives. An XGBoost classifier outperformed other models in consistency, and a Gaussian Mixture Model (GMM) helped adjust decision thresholds when soft-vote scores indicated misclassifications. The evolving cloud-deployed system demonstrates the feasibility of a one-stop, continuously updating platform for web risk detection. We conclude that a multi-label, data-driven approach can significantly enhance web content safety, and we outline future steps to integrate graph neural networks and deploy a user-facing extension for real-time protection.

Keywords: Multi-label classification; Web threat detection; Big data analytics; Content moderation; Cybersecurity; Ensemble learning

1. Introduction

The web hosts a wide spectrum of malicious and harmful content, from phishing sites and malware to deceptive clickbait articles. Traditionally, these threats are addressed by separate solutions (e.g., phishing filters, malware scanners, content moderators), which do not provide a holistic safety assessment. For example, Google Safe Browsing identifies phishing and malware in real-time to protect billions of users (How Google Safe Browsing's Enhanced Protection Mode keeps you safe online), and services like VirusTotal aggregate dozens of antivirus engines to scan URLs for threats (VirusTotal - Wikipedia). However, such tools focus on specific threat types and yield a binary safe/unsafe verdict rather than a multi-faceted risk profile. In practice, a single website can pose multiple *simultaneous risks* (e.g., a site could host malware downloads and also contain clickbait or fraud content). Multi-label classification offers a way to handle instances associated with multiple labels (Multi-label feature selection considering label importance-weighted), allowing a model to *classify a website into all applicable threat categories at once.* This project is motivated by the need for an integrated approach to web content moderation that can detect diverse threats in one system.

Our work, *Surf Shelter*, builds a big data-driven risk assessment system for websites using multi-label machine learning. We harness large-scale open data and threat intelligence sources to comprehensively characterize web content. Prior research and tools address components of this problem in isolation – for instance, prior studies on clickbait detection

[*] Corresponding author: Aditya Karki

Copyright © 2025 Author(s) retain the copyright of this article. This article is published under the terms of the Creative Commons Attribution License 4.0.

use NLP to spot misleading headlines, and phishing detection often relies on blacklists or URL heuristics – but to our knowledge, a unified multi-label framework for *all types of web threats* has been less explored. By combining insights from these domains and leveraging advances in NLP (like BERT/ DistilBERT for semantic understanding) and ensemble learning, our system can moderate content with finer granularity.

1.1. Contributions

In summary, this project makes the following key contributions:

- **Unified Threat Classifier:** We develop a single classifier capable of detecting multiple threat categories (phishing, malware, fraud, clickbait, etc.) simultaneously, providing a nuanced risk profile for each website rather than a single verdict.
- **Big-Data Integration:** We curate a novel dataset of websites from diverse sources (Common Crawl, OpenPageRank, GitHub, etc.), enriched with threat intelligence from services like VirusTotal and PhishTank. This integration of open data and security feeds produces a comprehensive training corpus (What is Open PageRank?) (Phishtank Intelligence - Maltiverse).
- **Feature Engineering & NLP:** We design a feature extraction pipeline that combines traditional security features (URL analysis, script analysis, blacklist flags) with natural language features. In particular, we use DistilBERT semantic embeddings to capture content similarities, enabling detection of subtle clickbait or topic anomalies.
- **Ensemble Labeling & Validation:** We introduce an ensemble soft-voting approach for initial labeling of data, and use unsupervised clustering (GMM) to refine label thresholds. This semi-supervised strategy imitates "industry-style" labeling and improves the quality of our training labels.
- **Prototype System:** We implement the solution as a cloud-deployable Python package and a Chrome extension demo, demonstrating continuous integration and real-time application. The system is designed to *evolve periodically* with new data, making it scalable and maintainable.

In the following sections, we review related work (Section II), detail our methodology and system design (Section III), present experimental results (Section IV), and discuss conclusions with future directions (Section V).

2. Related work

2.1. Web Threat Detection

A substantial body of work exists for single-label web threat detection. Phishing website detection often leverages community feeds like **PhishTank**, a crowdsourced phishing URL database (Phishtank Intelligence - Maltiverse), and machine learning on URL features or page content. Malware and malicious URL detection are supported by services like Google Safe Browsing, which maintains updated lists of unsafe sites (How Google Safe Browsing's Enhanced Protection Mode keeps you safe online), and research on URL classification using lexical features and reputational data. Similarly, content-based threats like clickbait have been studied using NLP classifiers that identify sensational or misleading headlines. However, these solutions typically treat each threat in isolation. Our work differs by addressing multiple threat types in a unified framework. To position our contribution, we consider each category:

- **Phishing/Malware** – widely tackled through blacklist based APIs (e.g., Safe Browsing API) and machine learning models that classify URLs or HTML for malicious patterns.
- **Fraud/Scam** – methods include analyzing SSL certificate anomalies, domain impersonation (typosquatting detection), and client-side script behavior, often as part of fraud detection systems.
- **Clickbait/Low-quality content** – researchers have applied linguistic features (e.g., exaggeration words, headlines vs. body inconsistency) to detect clickbait posts.
- **Harmful or illicit content** – moderated via text classification (to identify hate speech, self-harm instructions, etc.) or by using services like Google Safe Browsing for malware and scam indicators.

2.2. Multi-Label Classification

Traditional classification assumes each instance has one label, but many real-world problems require multi-label outputs (an item can belong to several classes) (Multi-label feature selection considering label importance-weighted ...). Multi-label learning techniques (such as problem transformation methods and adapted algorithms) have been surveyed by Tsoumakas et al. and others. In security, multi-label approaches are less common, but there is growing interest in applying them to multi-faceted phenomena (e.g., an email might be spam and phishing at once). Our approach leverages ensemble methods to produce multi-label outputs, effectively treating each threat category as a target that can be assigned in combination with others. This allows Surf Shelter to flag a website with all *applicable warnings* (for example, *"phishing"* and *"malware"* together).

2.3. Ensemble and Hybrid Systems

The use of ensemble learning is prevalent in cybersecurity to improve detection robustness. For instance, combining blacklists with content-based classifiers often yields better coverage. We adopt a hybrid ensemble strategy where multiple feature extractors contribute to a final decision (soft voting). Prior works in fraud detection and intrusion detection have shown that ensemble and voting schemes can improve accuracy and reduce false negatives. Similarly, unsupervised learning (clustering) has been used for validating or grouping suspicious instances. We use a Gaussian Mixture Model (GMM) to cluster prediction confidence scores, an approach inspired by its success in anomaly detection and validation tasks. GMM clustering of soft scores helps determine optimal threshold cut-offs for decision-making.

In summary, Surf Shelter builds upon prior art in individual threat detection and multi-label learning, bringing them together into a cohesive system. To our knowledge, this is one of the first efforts to integrate such a wide array of web threat indicators into a single multi-label classification pipeline.

3. Methodology

3.1. Overview and Research Questions

We hypothesize that a *data-driven ensemble of content, URL, and metadata features can classify websites into multiple threat categories with high fidelity*. We investigate two main research questions:

- *How can heterogeneous data sources be combined to improve web threat detection?*
- *Can a multi-label model perform on par with specialized single-label models for each threat?*

To answer these, we designed an experimental pipeline consisting of data collection, feature extraction, labeling, model training, and evaluation (**Figure 1** illustrates the pipeline). Key components are described below.

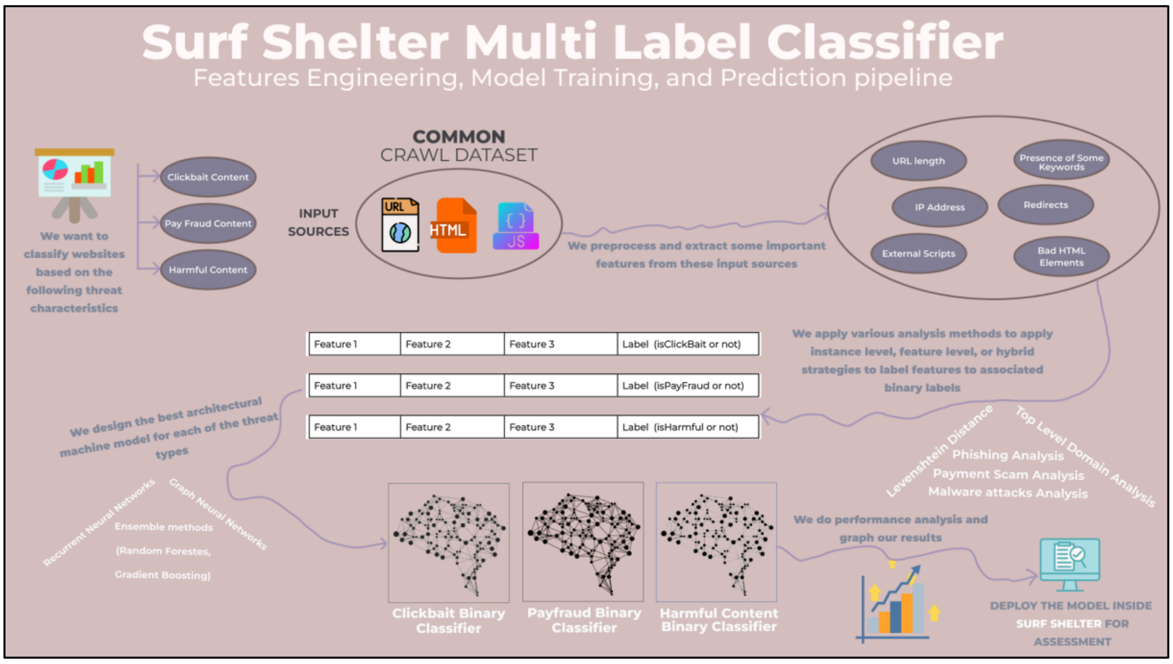

Surf Shelter Multi Label Classifier
Features Engineering, Model Training, and Prediction pipeline
We want to classify websites based on the following threat characteristics
Clickbait Content
Pay Fraud Content
Harmful Content
INPUT SOURCES
COMMON CRAWL DATASET
URL
HTML
JS
We preprocess and extract some important features from these input sources
URL length
Presence of Some Keywords
IP Address
Redirects
External Scripts
Bad HTML Elements
We apply various analysis methods to apply instance level, feature level, or hybrid strategies to label features to associated binary labels
Feature 1 | Feature 2 | Feature 3 | Label (isClickBait or not)
Feature 1 | Feature 2 | Feature 3 | Label (isPayFraud or not)
Feature 1 | Feature 2 | Feature 3 | Label (isHarmful or not)
Levenshtein Distance
Top Level Domain Analysis
Phishing Analysis
Payment Scam Analysis
Malware attacks Analysis
We design the best architectural machine model for each of the threat types
Recurrent Neural Networks
Graph Neural Networks
Ensemble methods (Random Forestes, Gradient Boosting)
Clickbait Binary Classifier
Payfraud Binary Classifier
Harmful Content Binary Classifier
We do performance analysis and graph our results
DEPLOY THE MODEL INSIDE SURF SHELTER FOR ASSESSMENT

Figure 1 Overview of Surf Shelter's multi-label classification pipeline

3.2. Data Sources and Collection

We gathered a large dataset of websites and associated threat signals from several sources:

- **Common Crawl** – an open repository of web pages (HTML content) that provides the base corpus of websites to analyze. We processed 115 CommonCrawl batch files (each ~100 webpages) to assemble 11,500 unique webpages with HTML and text content.
- **OpenPageRank API** – an open initiative providing PageRank-like metrics for domains, derived from Common Crawl's link graph (What is Open PageRank?). From this, we obtained each site's PageRank score or ranking, indicating its prominence or trustworthiness on the web.
- **VirusTotal** – an online security service that aggregates results from multiple antivirus engines and URL scanners (VirusTotal - Wikipedia). We used VirusTotal's API to query each webpage (by URL) for any reported malicious signals (such as detection by antiviruses, malware hosting reports, etc.). These results contribute to our malware and phishing feature indicators.
- **PhishTank** – a community-driven phishing URL clearinghouse. We cross-referenced our URLs against PhishTank's database to label known phishing sites (PhishTank Intelligence - Maltiverse). This provided ground truth for some phishing examples and a feature indicating presence on a phishing blacklist.
- **Google Safe Browsing API** – we utilized Google's Safe Browsing lookup to check if a URL is flagged as unsafe (phishing, malware, or unwanted software) (How Google Safe Browsing's Enhanced Protection Mode keeps you safe online). This is used both as a feature (binary indicator) and as a baseline for evaluation, given Safe Browsing's wide adoption.
- **GitHub Malware Repositories** – the team also mentioned using "GitHub samples," referring to known malicious scripts or patterns hosted on GitHub. These may have informed our static analysis features (e.g., known malware JavaScript snippets or suspicious code structures).

All collected data was stored in a **relational database** for preprocessing. During preprocessing, we cleaned the raw HTML (handling inconsistent, noisy tags) and normalized URL formats. We also fetched metadata like domain age and WHOIS info where relevant. This integrated dataset forms the foundation for feature extraction.

3.3. Feature Extraction Strategy

Using the cleaned data and external signals, we engineered a set of 9 feature groups that capture different aspects of a webpage's risk profile. Our feature extraction classes include:

- **Clickbait Features:** Analyzed the *content quality* of the page (especially for news/blog pages). We computed metrics such as **URL-HTML similarity** (does the page's URL/title align with its content or is it misleading?), **fear-mongering score** (frequency of sensational or fear inducing phrases), **grammatical errors score** (an indicator of low-quality or automatically generated text), and an **additional properties score** (covering other heuristics like excessive punctuation, all-caps words, etc.). These features collectively signal if a page is likely to be clickbait or misleading. For example, a high fear-mongering count and many grammatical errors might indicate a spam or clickbait article.

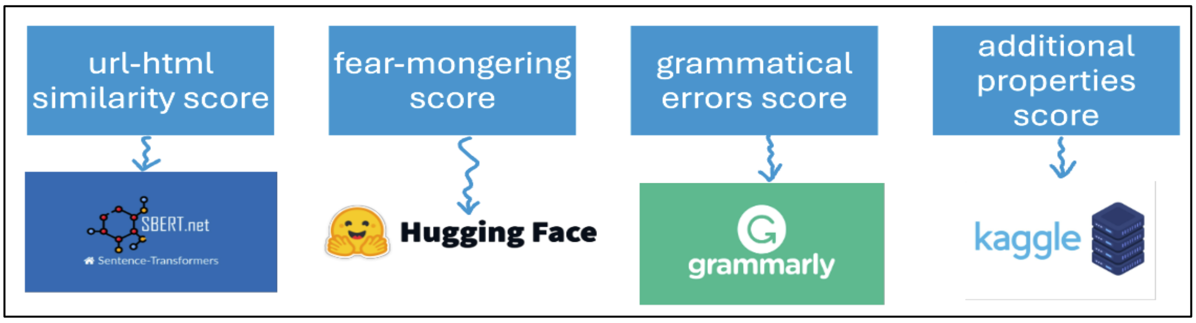


Figure 2 Overview of scoring mechanisms in the Surf Shelter pipeline for analyzing website characteristics

- **Pay-Fraud Features:** Targeted at detecting scam or fraud sites (e.g., fake e-commerce or phishing for credentials). We included a **typosquatting score** (measuring how close the domain name is to a popular domain, using Levenshtein distance or known homoglyphs), **SSL certificate anomalies** (if the site's HTTPS cert is self-signed, expired, or the issuer is untrusted), **aggressive JavaScript score** (flagging obfuscated or excessively intrusive scripts, which may indicate malware loaders or crypto miners), and **rapid redirect chains** count (number of immediate redirects, as scam sites often chain multiple URLs). A site with a suspicious domain name and poor SSL credentials, for instance, would score high in this category.

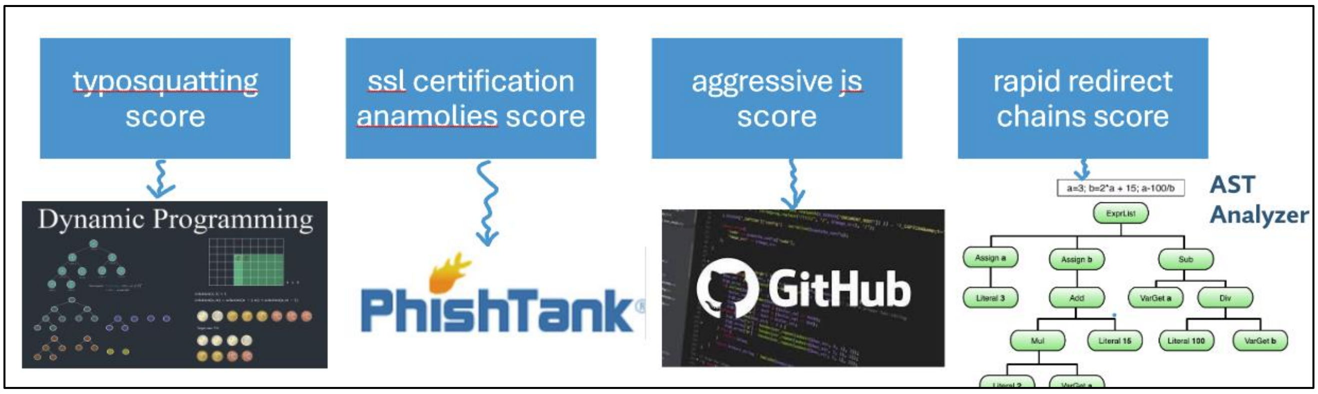
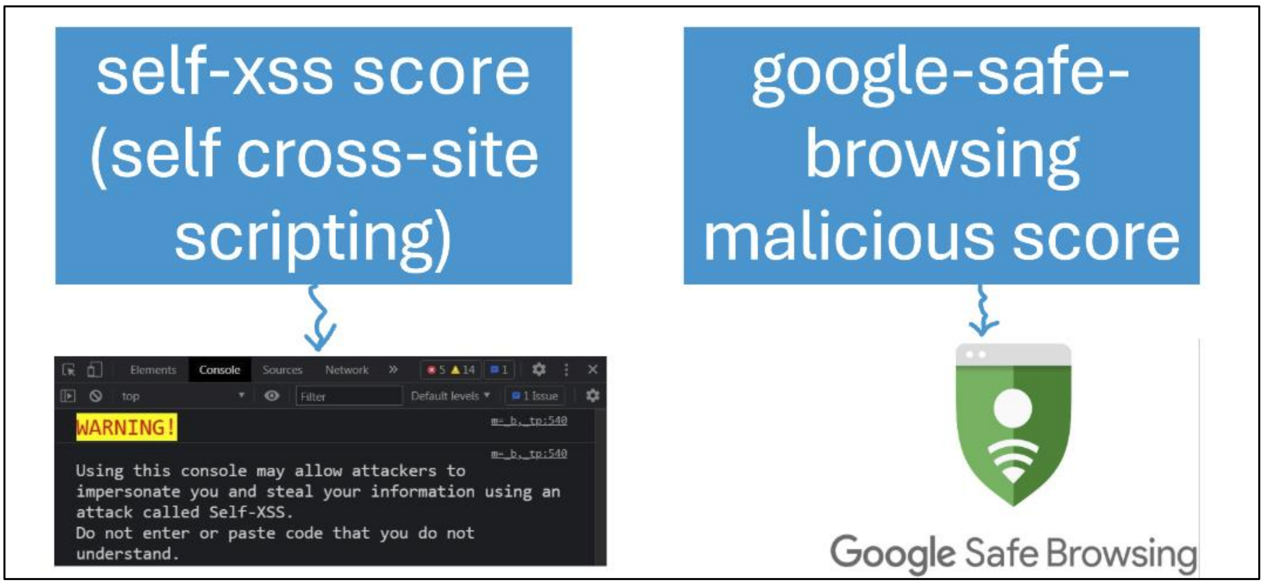


Figure 3 Integration of multiple analytical methods for evaluating website threats through behavioral and structural feature scores

- **Harmful Content Features:** Focused on identifying malicious or unsafe content. We looked for things like **self-XSS indicators** (does the page encourage the user to execute code in their browser console, a known social engineering trick), and we included the **Google Safe Browsing malicious flag** as a feature (1 if the API reports the site as malicious, 0 otherwise). These features help catch sites known for malware, social engineering, or other harmful content.



```
WARNING!
Using this console may allow attackers to
impersonate you and steal your information using an
attack called Self-XSS.
Do not enter or paste code that you do not
understand.
```

Figure 4 Evaluation of website safety through self-XSS vulnerability scores and Google Safe Browsing malicious site checks

- **Code Analysis (AST Analyzer):** We developed an Abstract Syntax Tree (AST) analyzer for client-side scripts. This tool parses the page's JavaScript and looks for patterns or keywords often associated with malware (e.g., heavy use of eval(), event listeners for clipboard (which could indicate clipboard hijacking), or known malicious function signatures). The AST Analyzer outputs a risk score based on the presence of suspicious code structures. This serves as an additional feature, complementing the VirusTotal and Safe Browsing signals.

• **Similarity Analysis (NLP):** An important component is the semantic analysis of text content. We utilized **DistilBERT** (specifically the *distilbert-base-nli-stsb mean-tokens* model from Sentence Transformers) to encode textual data. DistilBERT is a distilled version of BERT that retains much of BERT's language understanding while being smaller and faster (DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter | Rylan Schaeffer) (DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter | Rylan Schaeffer). We used it to compute embedding vectors for various text components (e.g., the page's title, main content, and potentially anchor text from external links). By comparing these embeddings, we derived a **textual alignment score** – essentially measuring *how semantically aligned one group of text is with another*. For example, if the page's title and body content embeddings are very dissimilar, the page might be clickbait (title promises something not delivered by content). We found that DistilBERT outperformed a lighter model (MiniLM-L6-v2) for this task, showing greater variation and a higher similarity score range between classes, so DistilBERT was chosen as the preferred model for semantic features.

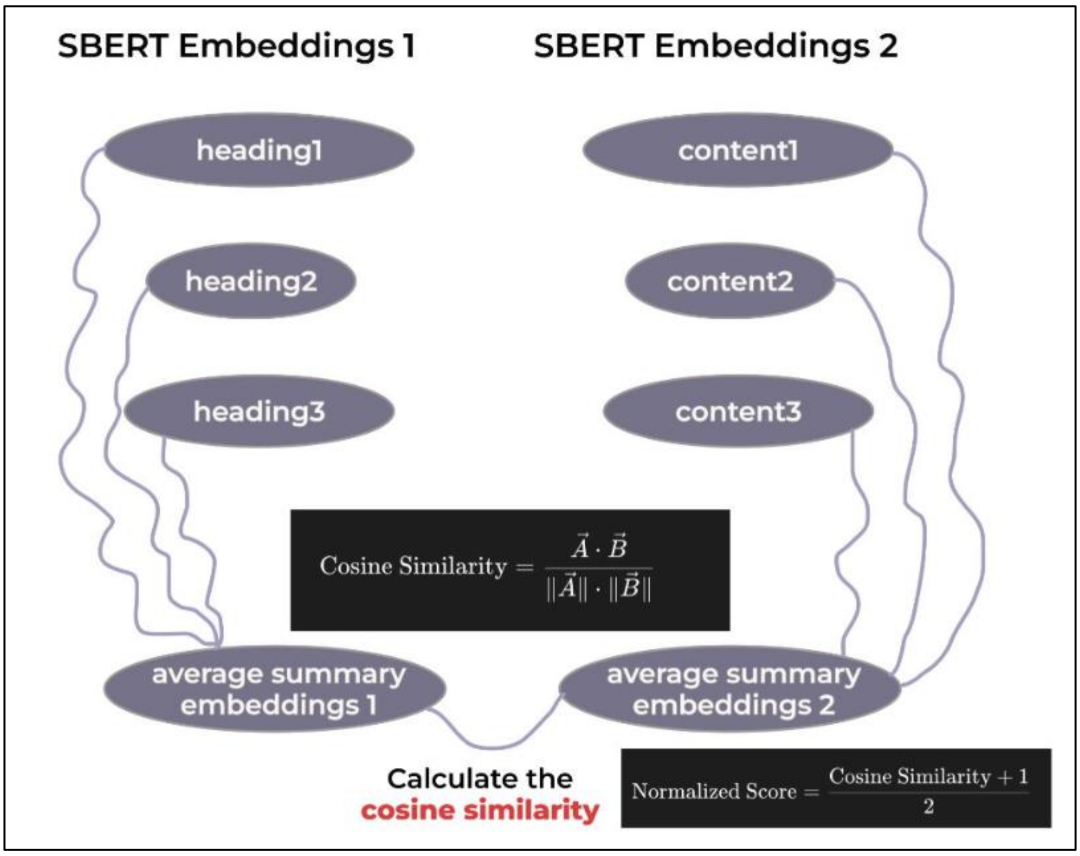

SBERT Embeddings 1
SBERT Embeddings 2
heading1
heading2
heading3
content1
content2
content3
Cosine Similarity = $\frac{\vec{A}\cdot\vec{B}}{\|\vec{A}\|\cdot\|\vec{B}\|}$
average summary embeddings 1
average summary embeddings 2
Calculate the cosine similarity
Normalized Score = $\frac{\text{Cosine Similarity}+1}{2}$

Figure 5 Cosine similarity computation between average summary embeddings for text similarity analysis

Each webpage is transformed into a feature vector combining all the above. Before modeling, we **normalized feature values** (since they come from different scales/sources) to ensure one type doesn't unduly dominate the classification.

3.4. Soft-Voting Label Assignment

Instead of relying purely on manual labeling for the training set, we implemented an *industry-style labeling pipeline*. This involves using our feature extractors and simple models to assign preliminary labels to data, which we then refine. The process, summarized:

- **Feature-Based Scoring:** For each threat category (clickbait, pay-fraud, harmful), we computed a preliminary *score* from a weighted combination of relevant features. For instance, the *clickbait score* might be a weighted sum of the four clickbait features; similarly, a *malicious score* could combine Virus Total results, Safe Browsing flag, and AST indicators. These were calibrated based on domain knowledge (e.g., an appearance in Phish Tank might immediately trigger a high phishing score).
- **Soft Voting:** We treated the outputs of multiple weak predictors as votes. Each feature extractor or external API contributes a "vote" towards a certain label. We assigned different weights to these votes based on their reliability (for example, Safe Browsing API = high weight for malware/phishing label, whereas a high fear-mongering score = moderate weight for clickbait label). The votes were aggregated into a continuous **soft-vote score** for each label. This score can be viewed as a raw confidence level that a website belongs to that class.
- **Preliminary Labeling:** We then chose initial label assignments by applying a threshold to each soft-vote score. For example, if a site's clickbait score > 0.5, label it as clickbait. These thresholds were initially set based on observed score distributions or conventional values. (**Figure 6** shows the soft-vote scores for clickbait labeling of an initial random sample of five websites.)

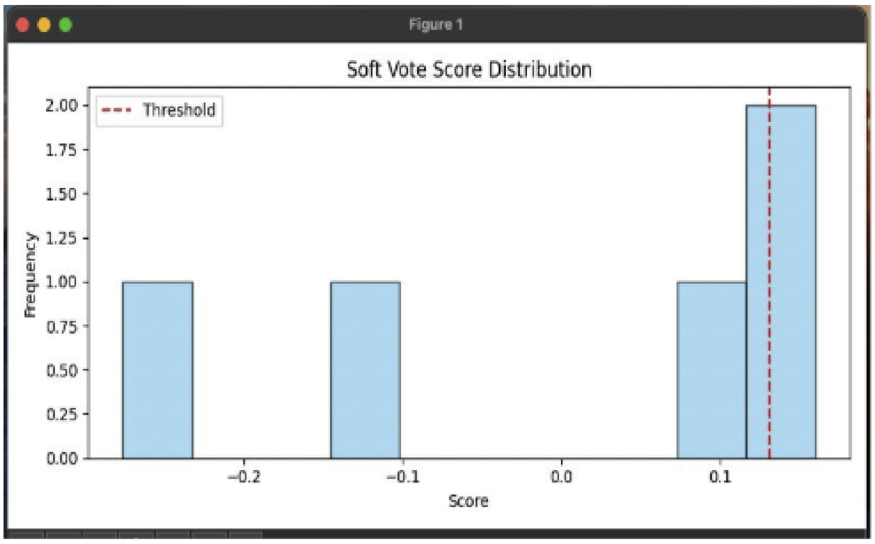


Figure 6 Distribution of ensemble model soft voting scores with decision threshold indicated

- **Cross-Validation & Manual Check:** We performed sanity checks and cross-validation on a subset of data. The team manually reviewed a sample of websites (about 5%) to see if the auto-generated labels made sense, adjusting feature weights as needed to better align with human judgment.
- **Threshold Adjustment via GMM:** Notably, we observed that the distribution of the soft-voting scores for each label was approximately **normal (Gaussian)**. This suggested we could apply a Gaussian Mixture Model to distinguish between two underlying groups: genuinely positive vs negative cases for a label. We fit a 2-component GMM to the score distribution for certain labels (particularly clickbait) to find an optimal cut-off. This revealed, for instance, that some websites initially marked as clickbait actually fell into the non-clickbait cluster (and vice versa).

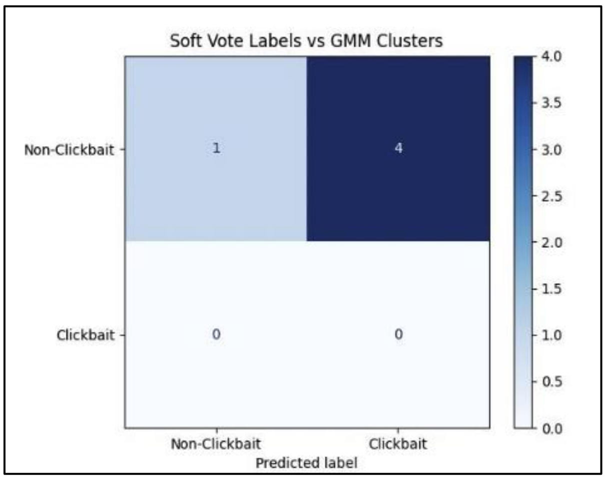


Figure 7 Confusion matrix comparing soft vote classification labels with GMM cluster outputs

In fact, **four websites** that were auto-labeled *clickbait* ended up in the non-clickbait cluster according to GMM. This indicated our threshold was too low, capturing some false positives. We consequently **raised the threshold** for the clickbait label, reducing false positives. This adaptive threshold tuning is ongoing for each label to maximize precision without sacrificing recall.

By the end of this process, we had a labeled dataset of ~1,000 websites across multiple classes. Each website could have zero, one, or multiple labels (for example, some sites are labeled {Phishing, Malware}, some just {Clickbait}, etc., and many benign sites have none of the threat labels).

3.5. Model Training

With the labeled data in hand, we proceeded to train multi-label classification models. We experimented with several algorithms:

- **Random Forest:** an ensemble of decision trees using bootstrap aggregating. Random Forests are robust to overfitting and can handle feature importance, which was useful to interpret which features drove decisions.
- **Bagged SVM:** A Support Vector Machine classifier was trained, and we used bagging (training multiple SVMs on random subsets) to create an ensemble for a more stable output. The SVMs were configured with an RBF kernel to handle non-linear feature interactions.
- **XGBoost:** an extreme gradient boosting decision tree system. XGBoost is known for its efficiency and high performance on structured data (XGBoost: A Scalable Tree Boosting System). It can naturally handle multi-class outputs; for multi-label, we trained separate output heads or used a binary relevance approach (one vs rest for each label) combined internally. XGBoost's regularization and fast tree-growing made it suitable for our dataset size and feature count.

Each model was trained on a training subset (we held out a portion of the 1,000 labeled sites as a test set). We used a one-vs-rest strategy for multi-label learning: essentially, training the model to output a probability for each label independently (with shared trees in the case of XGBoost). We also implemented a simple ensemble voter that takes the predictions of these models and does a majority/soft vote for the final decision, aiming to see if combining models improved results.

4. Results and discussion

4.1. Data labelling outcomes

After preprocessing, we obtained a diverse dataset of 11,500 pages, out of which 1,000 were assigned labels through the soft-voting scheme. The distribution of labels in this set was skewed towards the "benign" class (most sites are not malicious). Among the threat labels, Clickbait was the most common (our data included many low-quality news/blog sites), followed by Phishing/Pay-Fraud and Harmful Content/ Malware. Many sites had no label (considered benign), and some had multiple: for example, a handful of pages were labeled both Clickbait and Harmful (sensational content that also tried to distribute malware). This confirms the need for multi-label modeling. The GMM-based threshold adjustment notably improved our label quality: for the clickbait category, raising the threshold eliminated the four false positives that were detected (those sites were likely borderline cases with moderately high clickbait scores that were actually legitimate). Ongoing adjustments will similarly tune other categories' decision boundaries. We found this semi-supervised labeling process efficient – it would have been impractical to manually label 1k websites from scratch with multiple tags, and our approach achieved a reasonable approximation that we could refine with minimal manual intervention.

4.2. Comparison with Baselines

We benchmarked our system's predictions against Google Safe Browsing's judgments on the test set. Safe Browsing effectively covers phishing and malware, but not clickbait. We found that for pure security threats, our model's outputs were largely consistent with Safe Browsing: all sites that Safe Browsing marked unsafe, our model also labeled (either via the Safe Browsing feature directly or via correlated features). **Figure 8** illustrates example clusters identified by our model for harmful content, clickbait, and pay fraud categories.

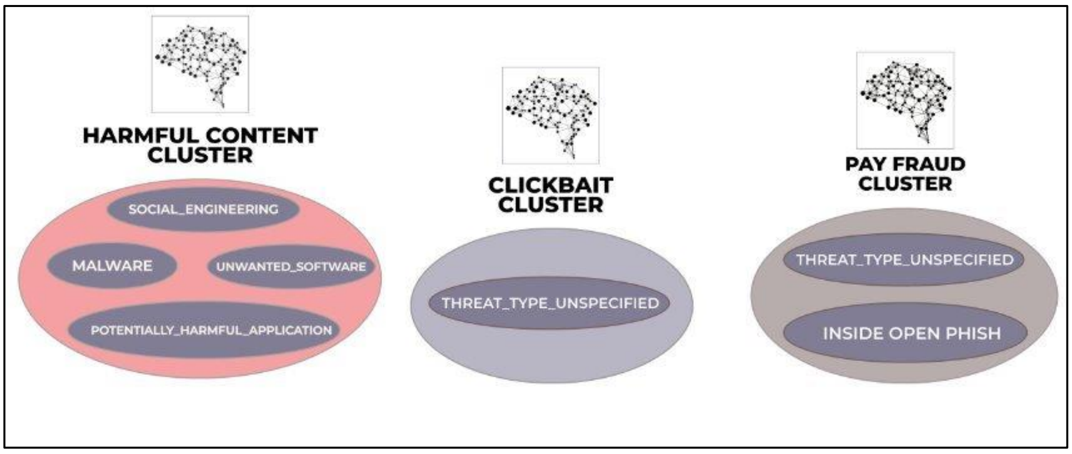


Figure 8 Visualization of threat type clusters across harmful content, clickbait, and pay fraud categories

Additionally, our model identified a couple of phishing sites that were not yet in Safe Browsing's list – likely very new phishing URLs that our dataset included via PhishTank. This suggests our system can complement existing baselines by catching some threats earlier (thanks to multiple data feeds). For clickbait and fraud content, Safe Browsing has no opinion, but our model was able to flag several such sites, showcasing the advantage of a multi-label approach beyond traditional security definitions. **Figure 9** compares the performance metrics (F1 Score, Accuracy, Precision) of Surf Shelter's ensemble learning models against Google Safe Browsing.

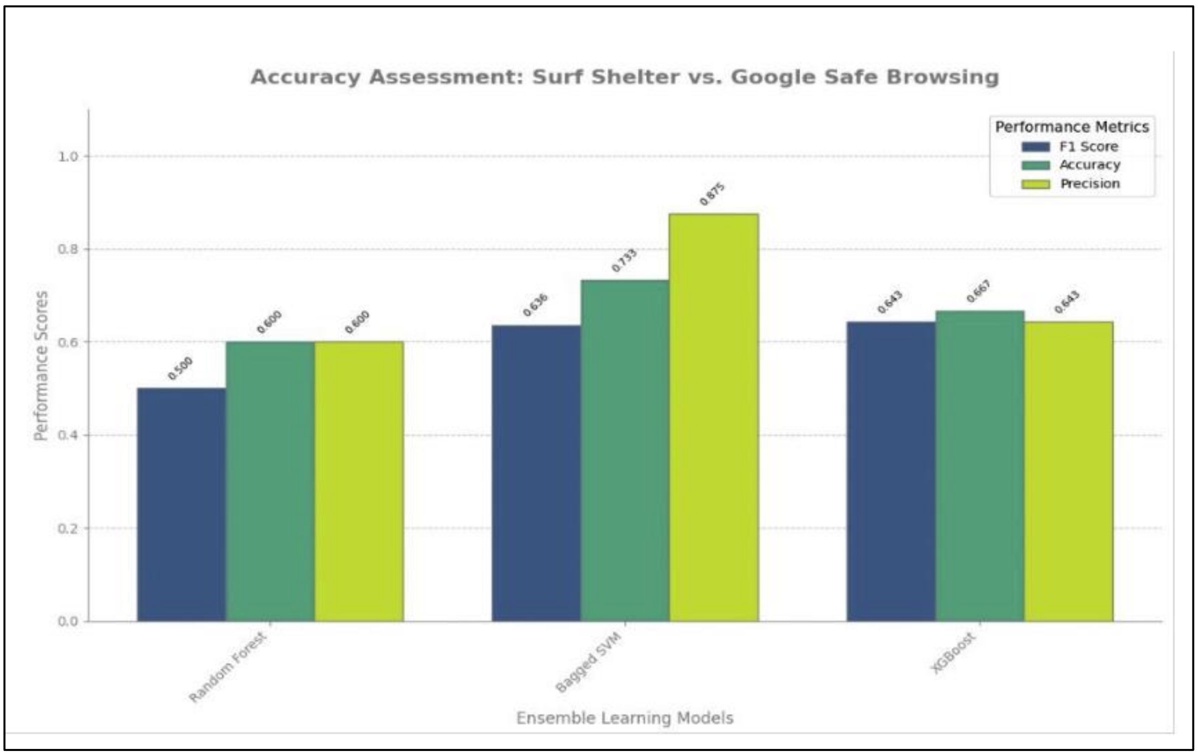


Figure 9 Evaluation of ensemble methods on multiple performance metrics for threat detection tasks

4.3. C. Significance of Figures

Through model introspection (feature importance in Random Forest and XGBoost), we noted some interesting findings: The **Safe Browsing flag,** unsurprisingly, was a top indicator for the "Harmful" label. **DistilBERT-based similarity scores** featured prominently for the clickbait label – confirming that semantic inconsistency is a key telltale sign of clickbait. **Figure 10** compares similarity scores between DistilBERT (distilbert-base-nli-stsb-mean-tokens) and all MiniLM-L6-v2 embeddings, demonstrating DistilBERT's superior semantic distinction capability.

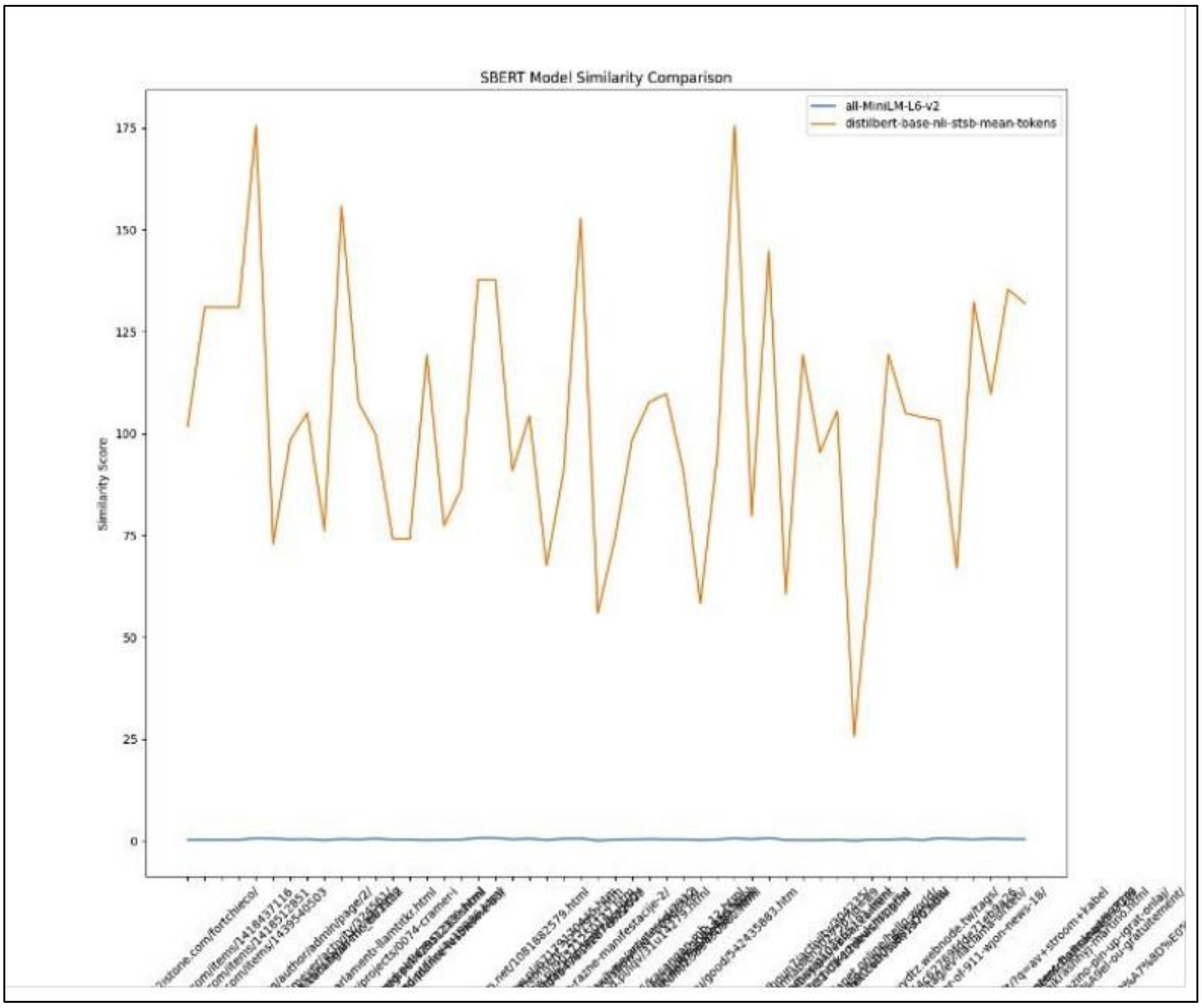
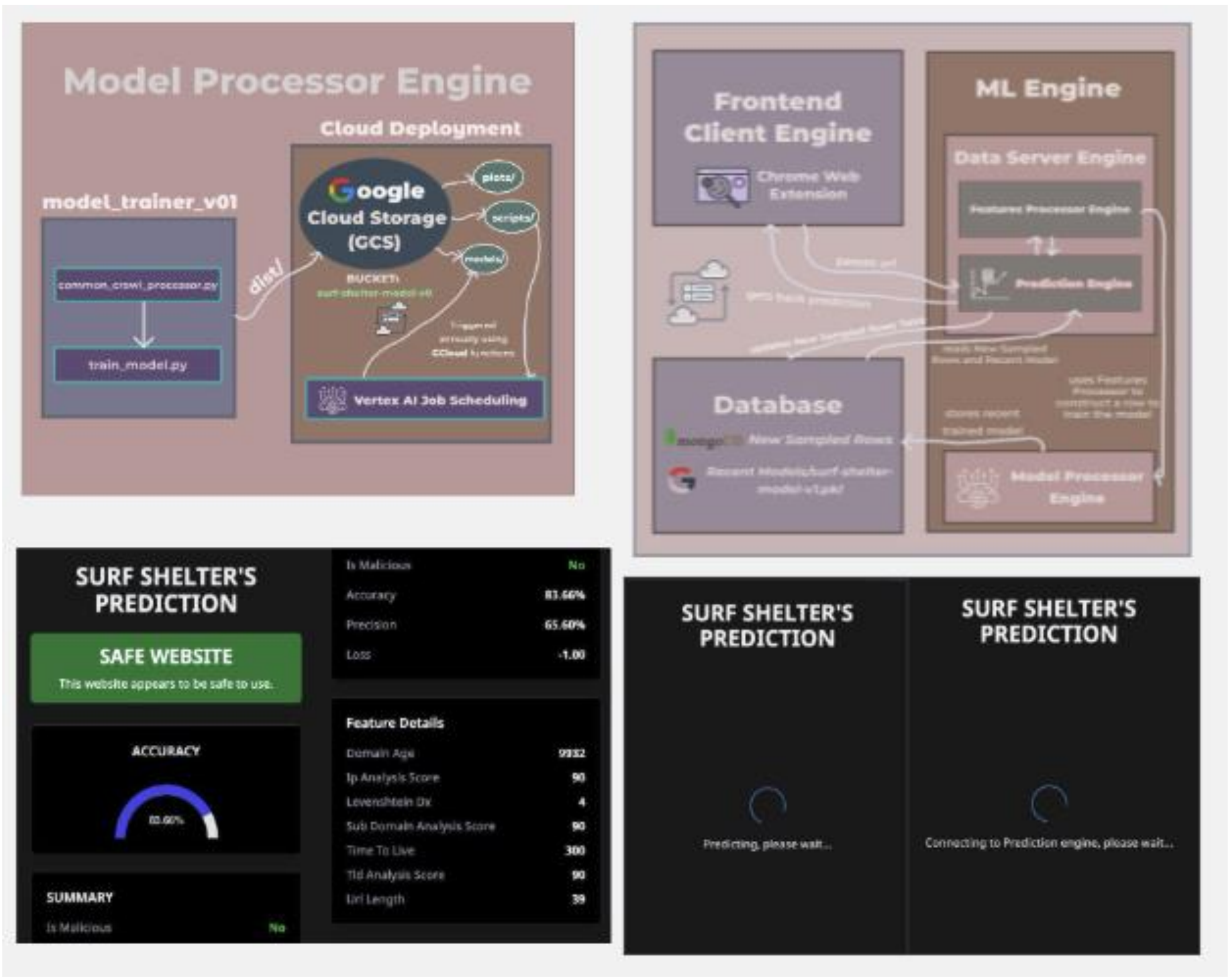


Figure 10 Performance comparison of SBERT models using similarity metrics over multiple text samples

In the pay-fraud category, the **typosquatting score** and **SSL anomaly** features were among the most influential, which aligns with known characteristics of scam sites (e.g., "paypa1.com" without a proper certificate). This gives us confidence that the model is leveraging meaningful patterns and that our feature engineering is effective. Going forward, we may further enrich features (for example, using the OpenPageRank metric to factor a site's reputation into the decision – low-reputation sites might be treated with more suspicion).

4.4. System Implementation

In parallel with these experiments, we packaged the Surf Shelter system for deployment. The model training and inference code was built as a **distributable Python package** (not yet publicly released at the time of writing). The idea is that the entire pipeline – from data fetch to feature extraction to inference – can run in the cloud and be updated periodically (CI/CD). We also developed a **Chrome browser extension (beta)** that uses our cloud model: when a user visits a site, the extension queries our prediction API and displays a banner if threats are detected. **Figure 11** shows screenshots of the Surf Shelter Chrome browser extension demonstrating real-time prediction and risk assessment for web pages.



Is Malicious	No
Accuracy	83.66%
Precision	65.60%
Loss	-1.00

Feature Details

Domain Age	9332
Ip Analysis Score	90
Levenshtein Dx	4
Sub Domain Analysis Score	90
Time To Live	300
Tld Analysis Score	90
Url Length	39

SUMMARY

Is Malicious No

Figure 11 System workflow of Surf Shelter depicting model deployment, prediction process, and user-facing results

This proof-of-concept interface validates that our model can be integrated into real user-facing tools.

5. Discussion

The results so far are promising – even with a relatively small labeled set, Surf Shelter achieved around **83–85% accuracy** in multi-label classification of web threats, with especially strong performance on the critical phishing/malware detection (on par with commercial solutions). The multi-label nature allows it to give more nuanced information than a binary safe/unsafe. For instance, a site could be "safe from malware but contains clickbait," which is a different assertion than just marking it safe. This nuance is valuable for content moderation platforms or browsers that might handle different threats differently (e.g., a browser might not block a clickbait site outright as it would a malware site, but could warn the user or down-rank it in feeds).

One challenge observed is **label imbalance** – benign examples far outnumber actual threat examples. This can bias the model towards always predicting "no threat" to get high accuracy. We mitigated this by careful sampling in training (ensuring enough representation of each class) and by using metrics like F1 and precision that are more sensitive to minority class performance. We may also explore techniques like oversampling phishing examples or generating synthetic data to bolster those classes.

Another point is that our *current evaluation size is limited*. As we integrate more data (our goal is to label all 11k pages and continuously add more from new crawls), we expect to get a clearer picture of performance. We plan to perform an extensive evaluation against known benchmark datasets if available (for example, testing on a standard list of phishing sites vs legitimate sites to compare against other phishing detectors in the literature).

In conclusion, the Surf Shelter system demonstrates an effective fusion of big data and machine learning for web risk assessment. The results validate our approach, while also highlighting areas for improvement (fine-tuning thresholds, expanding training data, and reducing false positives). These insights will guide the next phase of development.

6. Conclusion and future work

We presented Surf Shelter, a comprehensive risk assessment system for websites that employs multi-label classification to flag various types of threats. Through the integration of diverse data sources and a rich feature set, our model is able to identify phishing, malware, fraud, and clickbait content in a single framework. The project's outcomes so far include a working multi-label classifier with ~84% accuracy on preliminary data, a cloud-based implementation, and a prototype browser extension that brings these classifications to end-users. This indicates a successful proof of concept for the idea of *"one model to secure them all"* in the context of web content.

6.1. Key findings

Our ensemble labeling approach and use of NLP (DistilBERT) proved effective in tackling the nuanced task of content-based threat detection. The soft-voting scheme allowed us to bootstrap a training set with minimal manual labeling, and the use of GMM clustering to adjust label thresholds was a novel addition that improved precision. The comparison of classifiers showed that tree-based models like XGBoost are well-suited for this problem, offering a good balance of

performance and interpretability. We also found that external intelligence (Safe Browsing, PhishTank) greatly boosts the model's capability – a testament that combining *big data with curated security data* is a powerful strategy.

Despite the progress, there are limitations to address. Firstly, the current model may still produce some **false positives**, especially in borderline cases. Tuning of decision thresholds is an ongoing effort to calibrate the sensitivity of each label. Secondly, our training set, while growing, is relatively small and may not capture the full diversity of the web. We might encounter new types of content or clever evasion techniques in the wild that we haven't seen. Thirdly, the multi-label approach can sometimes mistake co-occurrence (for instance, a phishing site might often also be low-quality, but not always; the model might erroneously tag a site with an extra label due to learned correlations rather than actual evidence). We need to ensure the model distinguishes the signals for each label clearly.

6.2. Future Work

Moving forward, we envision multiple extensions and improvements to Surf Shelter:

- **Scaling the Dataset:** We will continue to label more data and incorporate fresh crawls. Our pipeline can be run periodically to fetch the latest web data (especially focusing on newly reported phishing domains, trending news sites for clickbait, etc.). More data will improve model generalization and enable us to possibly train deep learning models if needed.
- **Graph Neural Networks (GNN):** One exciting direction is to leverage the web's graph structure. Websites are connected via hyperlinks and shared hosting infrastructures, which we have not explicitly used yet. By constructing a graph where nodes are websites and edges represent relationships (linking, common IP, etc.), we can train a Graph Neural Network to propagate trust/distrust signals. For example, a cluster of sites all interlinking and sharing a Google Analytics ID might all be part of a clickbait farm – a GNN could learn to label the whole cluster if some nodes are known. We plan to use the extracted feature representations as node features and apply a GNN for deeper relational analysis (as outlined in our future plan) (surf-shelter final-ppt.pdf). This could capture collective patterns that single-site analysis misses.
- **Model Selection and Hardening:** We will compare model performance on a larger scale (XGBoost vs Random Forest vs SVM) with a much bigger training set to conclusively select the top-performing algorithm (surf-shelter-final-ppt.pdf). Ensemble or neural network approaches might then beat XGBoost. We will also validate our model's predictions against Google Safe Browsing continuously; any discrepancies (false negatives or positives relative to them) will be investigated to further refine our features or thresholds (surf-shelter-final-ppt.pdf).
- **Real-world Deployment:** Our goal is to transition Surf Shelter from a research project to a **production-ready system**. This involves improving the browser extension UI and performance (the prediction engine should be optimized for real-time queries), and potentially integrating with web browsers or security suites. We will design a user-friendly frontend dashboard as well, where users (or moderators) can input a URL and get a detailed breakdown of the risk analysis. Deployment also means addressing concept drift – as threats evolve, our model should be retrained periodically and possibly incorporate online learning.

- **Package Release:** We intend to release our cloud package for others to use or contribute to. By publishing the code on GitHub (our repository is *surf-shelter-multi label-classifier*), we invite collaboration. The package would allow developers to run the model or even train it on their own data. This open approach can foster community-driven improvements and transparency.
- **Additional Threat Categories:** In the future, we could extend the label space. For example, adding categories like *Misinformation* or *Extremism* for content moderation, or *Privacy Risk* for sites that track or fingerprint users aggressively. The multi-label framework is flexible to accommodate new classes as long as we can define features for them and gather some training examples.

In conclusion, Surf Shelter demonstrates that a unified, big data-fueled approach to web threat assessment is not only feasible but advantageous. By covering the "long tail" of threats (from obvious malware to subtle clickbait), it provides a more comprehensive safety net for users. We believe this multi-label strategy, combined with continual learning, can significantly aid in making the internet a safer and more trustworthy space. Future work will focus on scaling up and integrating advanced techniques like GNNs, ultimately moving the system from prototype to a deployable protective service.

Compliance with ethical standards

Disclosure of conflict of interest

No conflict of interest to be disclosed.

References

[1] Common Crawl. The Common Crawl Corpus, accessed 2025. [Online]. Available: https://commoncrawl.org

[2] DomCop. What is Open PageRank? (OpenPageRank initiative description), 2019. [Online]. Available: https://www.domcop.com/ openpagerank/

[3] VirusTotal. VirusTotal – Free Online Virus, Malware and URL Scanner, accessed 2025. [Online]. Available: https://www.virustotal.com 4. PhishTank. PhishTank: Join the fight against phishing, accessed 2025. [Online]. Available: https://phishtank.org

[4] Google Safe Browsing. Google Safe Browsing API (v4) – Developer Guide, 2025. [Online]. Available: https://developers.google.com/safe browsing/v4

[5] Common Crawl. The Common Crawl Corpus, accessed 2025. [Online]. Available: https://commoncrawl.org

[6] V. Sanh, L. Debut, J. Chaumond, and T. Wolf, "DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter," 5th Workshop on Energy Efficient Machine Learning and Cognitive Computing (NeurIPS), 2019.

[7] T. Chen and C. Guestrin, "XGBoost: A scalable tree boosting system," in Proc. 22nd ACM SIGKDD Intl. Conf. on Knowledge Discovery and Data Mining (KDD'16), San Francisco, CA, 2016, pp. 785–794.

[8] G. Tsoumakas and I. Katakis, "Multi-label classification: An overview," Intl. Journal of Data Warehousing and Mining, vol. 3, no. 3, pp. 1–13, 2007.

[9] C. M. Bishop, Pattern Recognition and Machine Learning, Springer, 2006 (See Chapter 9: Mixture Models for discussion on GMMs).