Research Article

Content Blocking Method To Reduce False Positives Based On Machine Learning

[1,]Andi Iksan Arkam, [2]Muhammad Yahya, [3,*]Abdul Wahid

[1,2,3] Department of Computer Engineering, the State University of Makassar, South Sulawesi, Indonesia

[*] Corresponding Author: wahid@unm.ac.id

Abstract: This study presents an experimental approach to enhance content-blocking systems by integrating machine learning with domain classification and Pi-hole DNS server technology. While traditional blocking mechanisms often result in false positives—legitimate domains mistakenly blocked—this research aims to mitigate such issues. By implementing various testing scenarios, including TF-IDF and N-gram feature extraction with and without preprocessing, the study evaluates the classification performance using the Naive Bayes algorithm. The results reveal the highest accuracy of 84% achieved with the N-gram method without preprocessing. This integrated approach shows promise in improving the precision of ad and website blocking mechanisms.

Keywords: False positives, pi-hole, machine learning, domain blocking, Content Blocking

Citation: Andi Iksan Arkam, Muhammad Yahya, & Abdul Wahid. (2025). Content blocking method to reduce false positives based on machine learning. *Iota*, 5(2). https://doi.org/10.31763/iota.v5i2.935

Academic Editor: Adi, P.D.P

Received: Maret 12, 2025

Accepted: April 23, 2025

Published: May 28, 2025

1. Introduction

The increasing implementation of technology has become a necessity across various sectors. These sectors include education, socio-cultural affairs, politics, economics, and others. One of the methods used to generate the information needed by society in the form of high-quality and relevant data is through technology. This data can be utilized for personal or public purposes. However, behind the sophistication of technology implementation lies an opportunity for malicious actors to exploit harmful domains, such as phishing practices, malware distribution, and other cyberattacks. Therefore, protecting internet users from dangerous domains is essential.

Using various techniques, cybercriminals create harmful advertisements, commonly referred to as malvertising, with the intent to disrupt or harm the consumers who view them. This is how cyber actors deliberately exploit the Internet advertising ecosystem for their gain (Arrate et al., 2020). Malvertising involves the distribution of malware, spyware, and other forms of cyberattacks through online advertisements that resemble legitimate ad content, potentially compromising users' operating systems, computers, and personal data (Studi et al., 2023).

The AdBlock method, which blocks malvertising during web browsing, can be used to prevent malware from spreading through ads. Although the implementation of AdBlock remains a topic of debate, many communities support this method by contributing large databases of harmful websites (Feal et al., 2021). Accidental access nowadays does not occur solely through keyword methods. There are other methods, such as using online advertisements embedded in websites, blogs, social media, games, and more. This is done by producers to attract consumer attention. However, not all embedded online advertisements contain positive elements; in fact, many contain negative content such as pornography, gambling, violence, and others (Sidik et al., 2023). Due to its ability to block tracking scripts such as AdTrack, Beacons, and Widgets depending on the configuration and blocking style, AdBlock is also often referred to as a website privacy tool.

Blocking ads at the DNS level is a common practice because it allows direct blocking of ads from the DNS in the area where the ads are located. This provides several benefits, such as easier network management and monitoring of activities within computer networks, and is expected to reduce internet advertisements with negative content and decrease complaints about intrusive ads when accessing websites, blogs, mobile applications, and other platforms (Sidik et al., 2023). Online ads that appear can be disruptive and are therefore blocked when users access the internet using DNS filtering. To prevent false positives when blocking domains, it is essential to maintain and regularly update a list of domains to be blocked with new entries so that ads using those domain names can also be stopped (Mujiastuti & Prasetyo, 2021).

Content blocking methods are a proactive approach to combat harmful content by restricting user access while browsing the internet. However, a key challenge is the increased potential for false positives—cases where safe domains are mistakenly classified as harmful and subsequently blocked. False positives can cause inconvenience for users, hinder normal access, and reduce trust in the blocking system. In this context, machine learning-based approaches have emerged as a promising solution to reduce false positives in domain-blocking methods. Machine learning enables the development of predictive models that can understand complex patterns in domain data and support more accurate decision-making.

This research was conducted in response to the increasing number of cyberattacks that exploit domains, website content, and advertisements as attack vectors—particularly in the form of social engineering, data theft, and ransomware distribution. Conventional methods, such as manually maintained blacklists or signature-based systems, often fail to keep pace with the rapid emergence of new threats. Therefore, machine learning-based approaches have become a promising solution, as they are capable of automatically recognizing harmful patterns and blocking malicious domains or content before they can harm users.

In this study, a machine learning-based blocking system is developed to detect and classify suspicious domains or content. This method is compared with conventional approaches such as blacklist filters, whitelists, and heuristic rule-based systems. Preliminary results show that the machine learning approach achieves higher accuracy and demonstrates better adaptability to emerging threats. This study aims to propose and develop a domain-blocking method that focuses on reducing false positives through a machine-learning approach. By integrating advanced feature analysis and appropriate classification methods, the objective of this research is to create an effective model for detecting harmful domains without unnecessarily disrupting access to safe ones.

2. Method

This study employs an experimental research approach, which examines the effects of a specific treatment or intervention on the observed subject. The research focuses on ad domain blocking, which is performed once to prevent ads from the same domain from appearing again. Therefore, it is necessary to regularly update the list of blocked domains to ensure that newly introduced ad domains are also blocked. In addition, maintenance is essential to reduce false positives, which are typically caused by an algorithm incorrectly identifying symptoms, signals, or objects that do not exist during the domain-blocking process. A machine learning-based approach to reducing false positives in domain blocking methods enables the development of predictive models capable of understanding complex patterns in-domain data.

This research was conducted at the State University of Makassar, specifically in the Network Computer Laboratory, located at Jalan Daeng Tata Raya, Parang Tambung Subdistrict, Tamalate District, Makassar City, South Sulawesi Province, 90224. The research period took place from November 2023 to February 2024. This study can also be conducted independently by the researcher, provided that the necessary facilities and infrastructure are fully available.

2.1 Design System

The research procedure includes the following stages: system requirements analysis, system design, system implementation, and system testing, The system requirements analysis or system design is shown in Figure 1. At this stage, an analysis is conducted to determine the necessary equipment required for this research, including both hardware and software components.

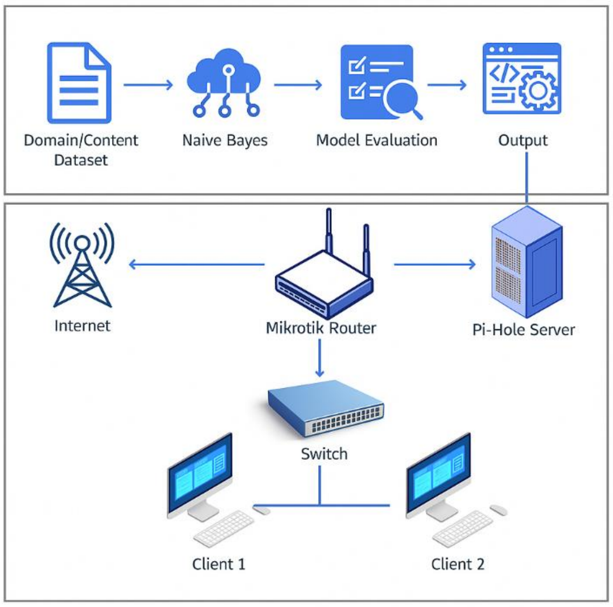


Figure 1. Machine Learning and Pi-Hole System Design

In Figure 1, when a client attempts to access a domain, the query is first received by the router and then forwarded to the ISP. The response from the ISP is returned to the router and subsequently passed to the Pi-Hole DNS for filtering. Pi-Hole filters the queries, determining which should be allowed and which should be blocked. After the filtering process, Pi-Hole sends the response back to the router, which then forwards it to the client's computer, delivering the query requested by the client. This process represents the implementation stage of machine learning.

Figure 2 illustrates the implementation stage of machine learning, where a dataset consisting of domain names is collected by the researcher. The labeled domain dataset undergoes a preprocessing stage and is then split to objectively assess the model's performance. The dataset is divided into training data, which is used during the model training process, and validation data, which is used to test the trained model. The next stage involves real-time testing to observe the results of feature extraction. Model evaluation is then conducted to assess and measure the model's performance based on the collected data.

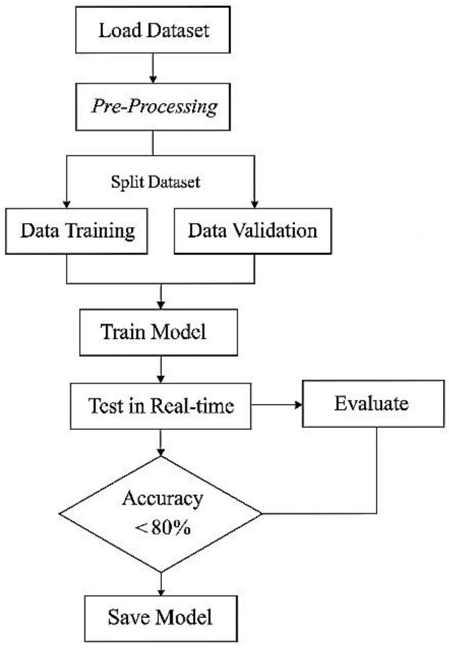


Figure 2. The implementation stage of machine learning

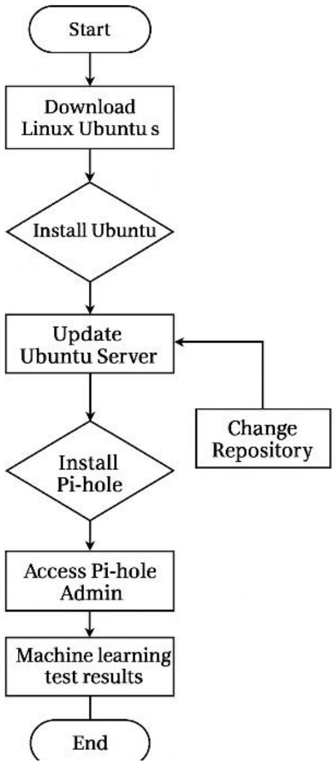


Figure 3. Steps in installing the Pi-Hole system

Figure 3 illustrates the steps involved in installing the Pi-Hole system. First, download and install Ubuntu Linux, then update the Ubuntu server. Once the update is complete, proceed with the installation of the Pi-Hole system, which functions as a tool to block website domains and ads on web pages based on the domain classification results. The final step is to configure the MikroTik router to provide local network access to client computers.

2.2 Data Collection

The data collection stages in this research are carried out in several steps, including Observation and Testing Scenarios. The observation in this study involved directly observing the object to obtain a record of advertising content domains while supporting data was gathered by reading and studying journals, articles, and books from various sources such as the internet and libraries related to this research. The research was conducted directly in the field by testing ad blocking based on the results of the machine learning model. Moreover, in this study, the DNS Server system will be tested for its ability to block advertisements and irrelevant domains on several websites. The testing was carried out using the black-box testing method, meaning the system is tested based on its inputs and outputs without the need to understand its internal implementation details. Domain samples were taken from the websites *kaggle.com* and *semrush.com*.

3. Result and Discussion

The results of this study summarize the entire process that has been carried out, including data collection, preprocessing, feature extraction, classification, model testing, evaluation, and System Test Results. Each subsection in this part will outline the key aspects of the research and provide a comprehensive understanding of the study's achievements.

3.1 Data Collection

The data collection process was carried out by gathering datasets from two sources: secondary data and primary data. The secondary data was obtained from the websites www.kaggle.com and www.semrush.com, while the primary data was collected directly by the researcher from websites. A total of 501 data entries were gathered. The dataset was compiled into a single CSV file in table format and labeled according to the condition of each entry. The dataset consists of three categories: web, porn, and ads. Therefore, the dataset labels are divided into two categories: false positive (1) and not false positive (0).

3.2 Preprocessing

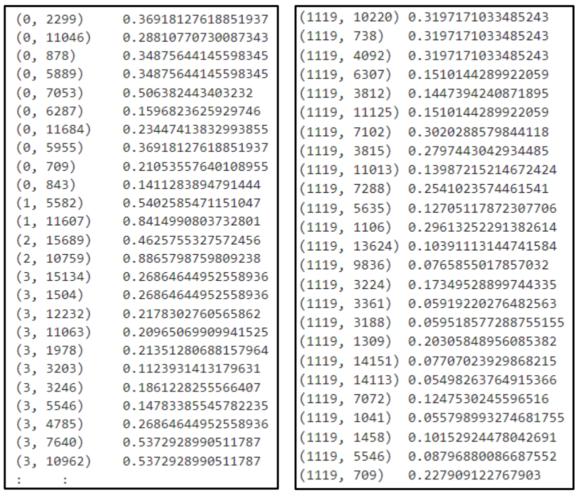
This study employs two scenarios in the preprocessing stage: with preprocessing and without preprocessing. The implementation of these two scenarios is conducted to compare the performance of the algorithm in the classification process with and without preprocessing. The collected data is then used in the model training process for the implementation of the Naïve Bayes algorithm. However, before proceeding to the Naïve Bayes implementation stage, a preprocessing step is applied to the data (Scenario 1). The preprocessing stage includes stop-word removal, case folding, stemming, and tokenization.

3.3 Split Dataset

The dataset is divided to assess the extent to which the model can be trained. The training data is used to train the algorithm to recognize patterns and characteristics within the domain dataset, which has been categorized into two classes: positive and negative. The testing data is used to evaluate the performance of the trained algorithm. The testing data serves as an independent evaluation set, where the algorithm is tested using data it has not encountered before. The test results are used to measure the accuracy and performance of the algorithm in classifying positive and negative domains.

3.4 *Feature Extraction*

This study employs two scenarios in the feature extraction stage: one using the TF-IDF method and the other using the N-gram method. The implementation of these two scenarios aims to compare the performance of the algorithm in the classification process by applying different feature extraction techniques. The first scenario is conducted by implementing the TF-IDF method in the feature extraction stage. The initial step in the TF-IDF feature extraction process involves importing the TfidfVectorizer class from the sklearn.feature_extraction.text module in the Scikit-learn library. The next step is converting the collection of words in each document into a numerical vector representation based on word frequency and inverse document frequency (IDF) weighting. This representation helps the algorithm classify domains as either false positive or negative. Figure 4.1 shows the results of the feature extraction stage using the TF-IDF method.



```
(0, 2299)     0.36918127618851937
(0, 11046)    0.28810770730087343
(0, 878)      0.34875644145598345
(0, 5889)     0.34875644145598345
(0, 7053)     0.506382443403232
(0, 6287)     0.1596823625929746
(0, 11684)    0.23447413832993855
(0, 5955)     0.36918127618851937
(0, 709)      0.21053557640108955
(0, 843)      0.1411283894791444
(1, 5582)     0.5402585471151047
(1, 11607)    0.8414990803732801
(2, 15689)    0.4625755327572456
(2, 10759)    0.8865798759809238
(3, 15134)    0.26864644952558936
(3, 1504)     0.26864644952558936
(3, 12232)    0.2178302760565862
(3, 11063)    0.20965069909941525
(3, 1978)     0.21351280688157964
(3, 3203)     0.1123931413179631
(3, 3246)     0.1861228255566407
(3, 5546)     0.14783385545782235
(3, 4785)     0.26864644952558936
(3, 7640)     0.5372928990511787
(3, 10962)    0.5372928990511787
:     :
```

```
(1119, 10220) 0.3197171033485243
(1119, 738)   0.3197171033485243
(1119, 4092)  0.3197171033485243
(1119, 6307)  0.1510144289922059
(1119, 3812)  0.1447394240871895
(1119, 11125) 0.1510144289922059
(1119, 7102)  0.3020288579844118
(1119, 3815)  0.2797443042934485
(1119, 11013) 0.13987215214672424
(1119, 7288)  0.2541023574461541
(1119, 5635)  0.12705117872307706
(1119, 1106)  0.29613252291382614
(1119, 13624) 0.10391113144741584
(1119, 9836)  0.0765855017857032
(1119, 3224)  0.17349528899744335
(1119, 3361)  0.05919220276482563
(1119, 3188)  0.059518577288755155
(1119, 1309)  0.20305848956085382
(1119, 14151) 0.07707023929868215
(1119, 14113) 0.05498263764915366
(1119, 7072)  0.1247530245596516
(1119, 1041)  0.055798993274681755
(1119, 1458)  0.10152924478042691
(1119, 5546)  0.08796880086687552
(1119, 709)   0.227909122767903
```

Figure 4. Feature extraction results using the N-gram method

In Figure 4 The explanation of the feature extraction results using the N-gram method is as follows:

a) Each row in the output represents one N-gram entity within a specific document.
b) The format of each row is '(document index, feature index) frequency value'.
c) The document index is the identification number for a particular document in the dataset.
d) The feature index refers to the index of the N-gram within the learned vocabulary.
e) The frequency value indicates how many times the N-gram appears in that specific document.
f) For example, '(0, 3877) 1' means that the N-gram with index 3877 appears once in the first document.

3.5 *Classification*

The implementation stage of the Naive Bayes algorithm for classifying false positive and negative domains is the core phase of this research. The process begins by importing the MultinomialNB class from the sklearn.naive_bayes module. MultinomialNB is a class that represents the Naive Bayes classification model applied to data with a multinomial distribution, such as text data. In the next step, the Naive Bayes classification model is trained using the training data that has been transformed into either a TF-IDF vector representation or an N-gram representation. The training data comprises 80% of the entire dataset. The resulting model is then used to make predictions on the testing data or new data that has a similar representation.

The parameters used in the domain classification testing with the Naive Bayes architecture are as follows:

a) Length is a parameter used to determine the length of a domain. For example, www.malicious-site-12345.com is more suspicious compared to a short and familiar domain like google.com.
b) A subdomain is a parameter used to identify the length or number of subdomains that imitate legitimate domains. For example, secure-login.bankofamerica.com.malicious.com is more suspicious than bankofamerica.com.
c) Characters are a parameter used to examine a domain from the perspective of the words or numbers contained within it. For example, pay-pal-login.com is more suspicious than paypal.com.

The Naive Bayes classification stage in this study was applied in four different scenarios. These scenarios involved combinations of either using or not using data preprocessing, along with different feature extraction methods. Specifically, the first scenario referred to the use of the TF-IDF feature extraction method with data preprocessing; the second scenario referred to the use of TF-IDF without data preprocessing; the third scenario involved the use of the N-gram feature extraction method with data preprocessing; and the fourth scenario involved the use of N-gram without data preprocessing. These four scenarios were implemented to evaluate the performance of each and to determine which scenario yielded the best results in data classification.

3.6 *Model Testing*

Model evaluation is a crucial stage that allows us to measure how well the trained model can generalize to data it has never seen before. The first research scenario, which involved using the TF-IDF feature extraction method with data preprocessing, showed an accuracy of 74%. Based on the evaluation results, it is necessary to improve the model's accuracy by applying the second, third, and fourth scenarios. The second scenario, which used the TF-IDF feature extraction method without data preprocessing, showed better accuracy compared to the first scenario, with an accuracy of 81%. Similar results were observed in the third and fourth scenarios, which achieved higher accuracy than the first scenario, with accuracy values of 78% and 84%, respectively.

3.7 *Evaluation*

Model evaluation is a critical stage that allows us to measure how well the trained model can generalize to previously unseen data. The first research scenario, which involves the use of the TF-IDF feature extraction method combined with data preprocessing, resulted in an accuracy of 74%. Based on the evaluation results, the model's accuracy needs to be improved to achieve better performance by applying the second, third, and fourth scenarios.

The second scenario, which uses the TF-IDF feature extraction method without data preprocessing, showed better performance compared to the first scenario, with an accuracy of 81%. Similar results were observed in the third and fourth scenarios, which also demonstrated improved accuracy over the first scenario, with accuracy scores of 78% and 84%, respectively.

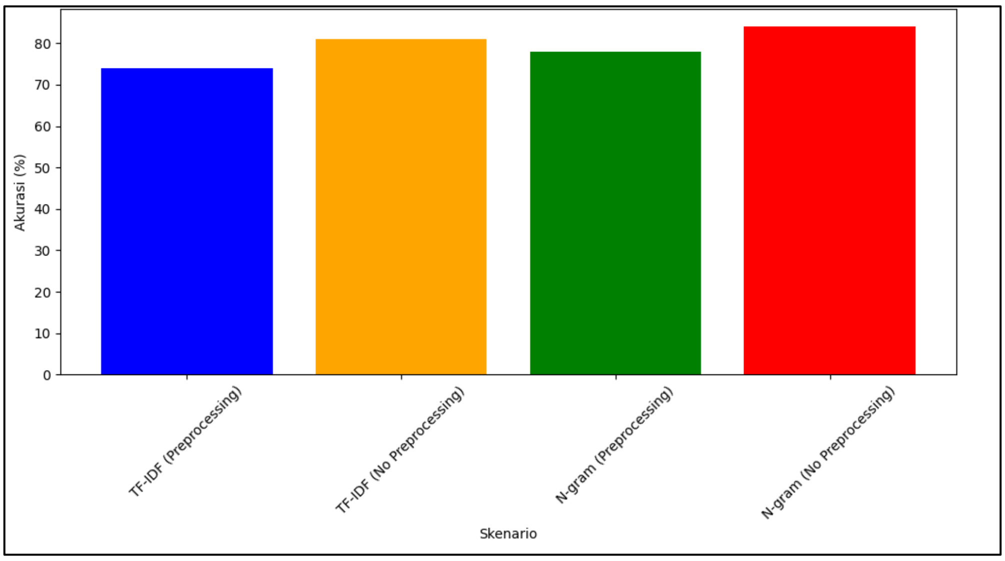
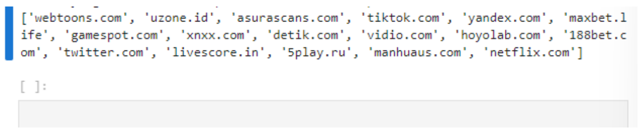


Figure 5. Accuracy of the fourth research scenario

Based on Figure 5, the fourth research scenario, which refers to the use of the N-gram feature extraction method without data preprocessing, shows the highest accuracy performance compared to the others, with an accuracy of 84%.

The first scenario demonstrated good performance, achieving an accuracy rate of 74% based on the total training dataset. This test was conducted using the TF-IDF feature extraction method with data preprocessing. The second scenario also showed good performance, with an accuracy rate of 81% from the total training dataset. This test was carried out using the TF-IDF feature extraction method without data preprocessing. The third scenario exhibited good performance as well, with an accuracy rate of 78% based on the total training dataset. This test involved the N-gram feature extraction method with data preprocessing. The fourth scenario achieved good performance, with the highest accuracy rate of 84% from the total training dataset. This test was conducted using the N-gram feature extraction method without data preprocessing. These four scenarios were implemented to evaluate the performance of each and to determine which scenario yielded the best results in data classification. Moreover,m Figure 6 shows the output results from the machine learning model predicting domains classified as false positives based on the model's predictions.



```
['webtoons.com', 'uzone.id', 'asurascans.com', 'tiktok.com', 'yandex.com', 'maxbet.l
ife', 'gamespot.com', 'xnxx.com', 'detik.com', 'vidio.com', 'hoyolab.com', '188bet.c
om', 'twitter.com', 'livescore.in', '5play.ru', 'manhuaus.com', 'netflix.com']

[ ]:
```

Figure 6. Output results from the machine learning model

3.8 *System Test Results*

The Pi-Hole DNS Server was tested under several scenarios, including website traffic testing without the system using Pi-Hole and Ad-Blocker, ad-blocking tests on websites, and domain-blocking tests. The system was evaluated for its ability to block ads (Ad-Blocker) and domains on several websites to determine whether the Pi-Hole DNS Server could successfully perform the blocking functions.

The ad block testing was conducted to determine whether Pi-Hole and Ad-Blocker are capable of blocking ads on accessed domains. The test was performed by comparing the same website before and after implementing Pi-Hole and Ad-Blocker. This test was conducted on websites that contain a large number of online advertisements, such as *kompas.com*, where ads were still displayed, as shown in the following Figure 7.

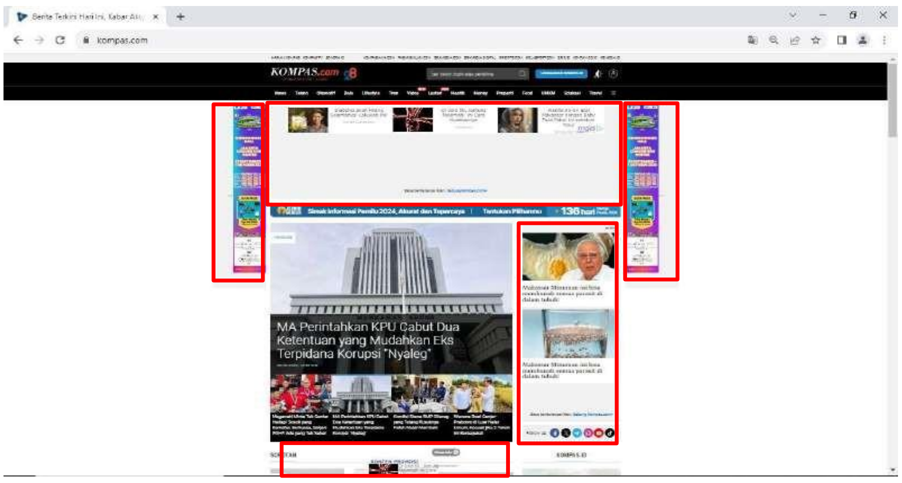

kompas.com
KOMPAS.com
MA Perintahkan KPU Cabut Dua Ketentuan yang Mudahkan Eks Terpidana Korupsi "Nyaleg"

Figure 7. The appearance of the kompas.com website without using the Pi-Hole

Figure 7 shows the appearance of the *kompas.com* website without using the Pi-Hole DNS Server and before activating the AdBlocker extension. The website displays numerous online advertisements on the homepage, including pop-up ads, side-page ads, and bottom-page ads. These advertisements can disrupt the user experience while browsing and contribute to slower website loading times.

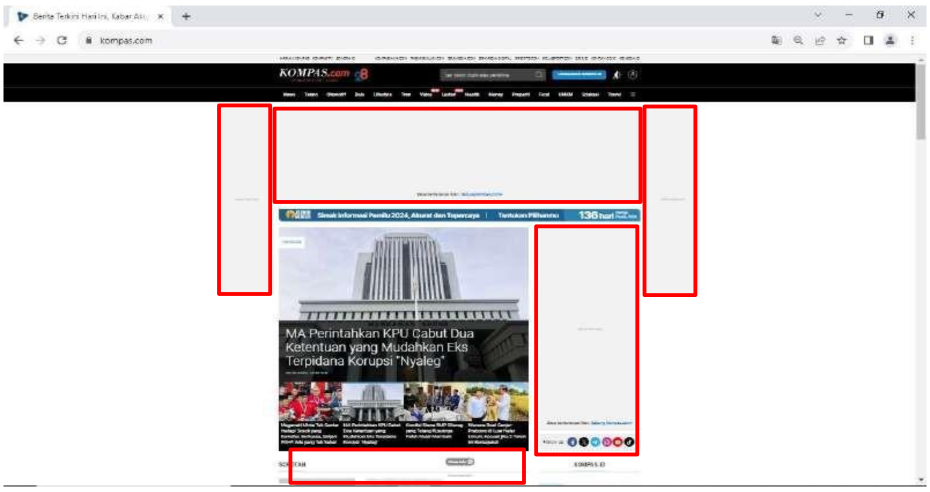

kompas.com
KOMPAS.com
MA Perintahkan KPU Cabut Dua Ketentuan yang Mudahkan Eks Terpidana Korupsi "Nyaleg"

Figure 8. The appearance of the kompas.com website after implementing the Pi-hole

Figure 8 shows the appearance of the *kompas.com* website after implementing the Pi-hole DNS Server and activating the AdBlocker extension. The website no longer displays advertisements on the homepage. Online ads that previously appeared as pop-ups, side-page ads, and bottom-page ads have been removed, resulting in an unobstructed browsing experience. Users accessing the website are no longer disturbed by advertisements, and the page loads significantly faster.

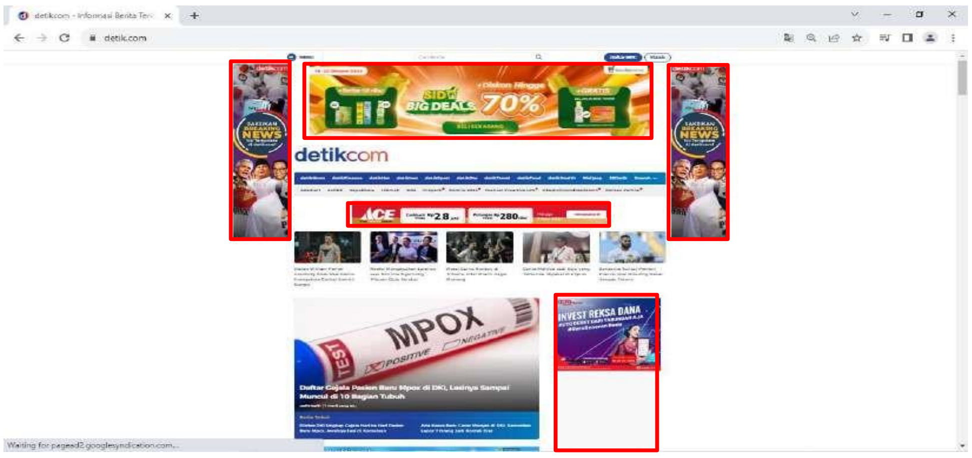


Figure 9. The appearance of the detik.com website before activating the AdBlocker extension

Figure 9 shows the appearance of the *detik.com* website before activating the AdBlocker extension. The website displays numerous online advertisements on the homepage, including pop-up and side-page ads. These ads can disrupt the user experience while browsing the site and slow down the website's loading time.

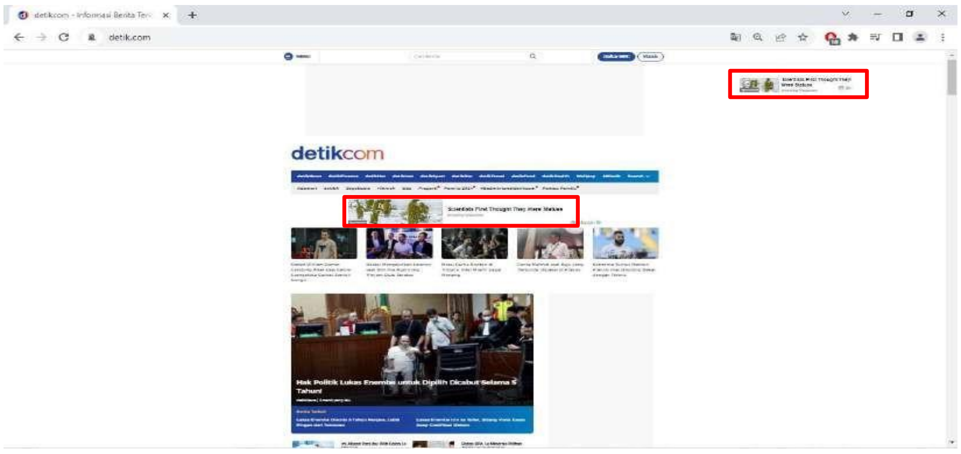
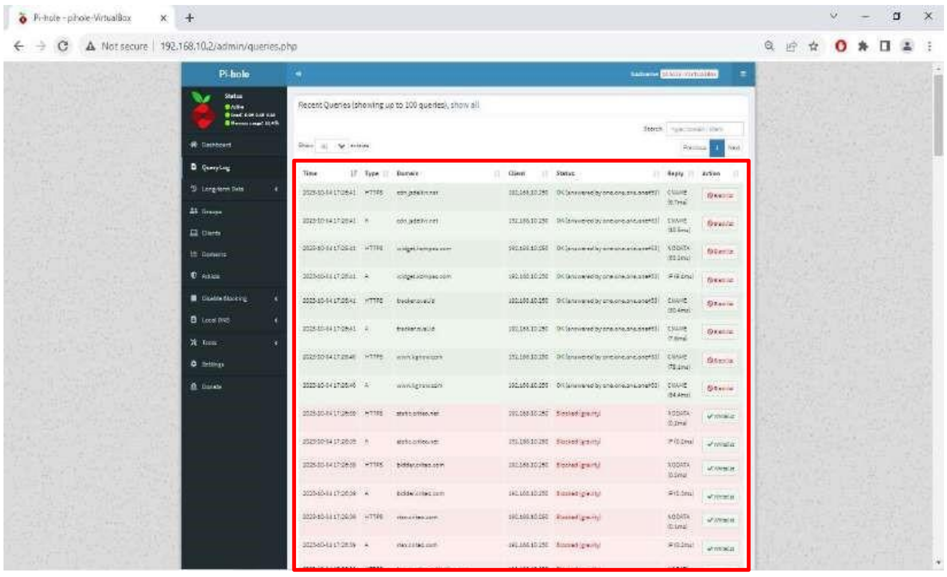


Figure 10. The appearance of the detik.com website after activating the AdBlocker extension

Figure 10 shows the appearance of the *detik.com* website after activating the AdBlocker extension. The site still displays some advertisements on the homepage. Using only AdBlocker to block ads proves to be inefficient, as ads continue to appear. Users accessing the website are still disturbed by the presence of online advertisements.



Pi-hole - pihole-VirtualBox

Not secure | 192.168.10.2/admin/queries.php

Recent Queries (showing up to 100 queries), show all

Time	Type	Domain	Client	Status	Reply	Action
2023-10-04 17:26:41	HTTPS	cdn.jsdelivr.net	192.168.10.250	OK (answered by one.one.one.one#53)	CNAME (8.7ms)	Blacklist
2023-10-04 17:26:41	A	cdn.jsdelivr.net	192.168.10.250	OK (answered by one.one.one.one#53)	CNAME (10.5ms)	Blacklist
2023-10-04 17:26:41	HTTPS	widget.kompas.com	192.168.10.250	OK (answered by one.one.one.one#53)	NODATA (33.1ms)	Blacklist
2023-10-04 17:26:41	A	widget.kompas.com	192.168.10.250	OK (answered by one.one.one.one#53)	IP (9.0ms)	Blacklist
2023-10-04 17:26:41	HTTPS	[illegible]	192.168.10.250	OK (answered by one.one.one.one#53)	CNAME (30.4ms)	Blacklist
2023-10-04 17:26:41	A	[illegible]	192.168.10.250	OK (answered by one.one.one.one#53)	CNAME (7.8ms)	Blacklist
2023-10-04 17:26:40	HTTPS	[illegible]	192.168.10.250	OK (answered by one.one.one.one#53)	CNAME (78.1ms)	Blacklist
2023-10-04 17:26:40	A	[illegible]	192.168.10.250	OK (answered by one.one.one.one#53)	CNAME (84.4ms)	Blacklist
2023-10-04 17:26:09	HTTPS	static.criteo.net	192.168.10.250	Blocked (gravity)	NODATA (0.2ms)	✓ Whitelist
2023-10-04 17:26:09	A	static.criteo.net	192.168.10.250	Blocked (gravity)	IP (0.2ms)	✓ Whitelist
2023-10-04 17:26:09	HTTPS	bidder.criteo.com	192.168.10.250	Blocked (gravity)	NODATA (0.5ms)	✓ Whitelist
2023-10-04 17:26:09	A	bidder.criteo.com	192.168.10.250	Blocked (gravity)	IP (0.5ms)	✓ Whitelist
[illegible]	HTTPS	[illegible]	[illegible]	Blocked (gravity)	NODATA	✓ Whitelist
[illegible]	A	[illegible]	[illegible]	Blocked (gravity)	IP	✓ Whitelist

Figure 11. The query log display on the Pi-Hole system

Figure 11 shows the query log display on the Pi-Hole system, which presents the monitoring results of websites accessed by the client. It also shows an additional blocked query for *kompas.com* when the client accesses the site. This test analysis was conducted to compare website traffic accessed by the client, presenting the test results both without the system and using Pi-Hole and Ad-Blocker. The sample domains were taken from the website semrush.com. Below are several domains that have been analyzed based on traffic comparison results on the websites.

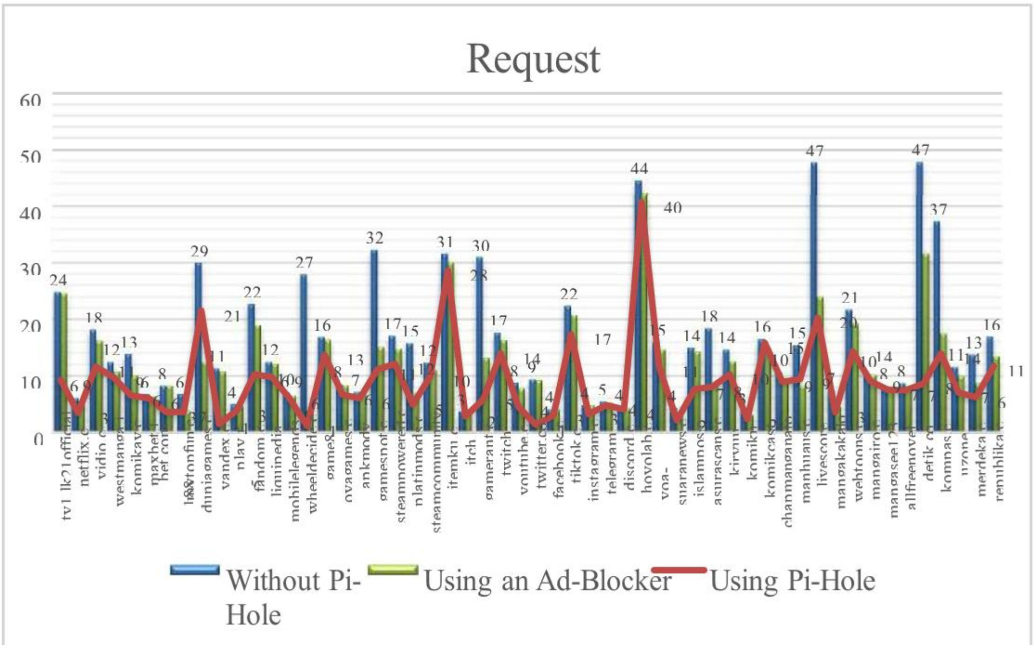


Figure 12. Request Data Graph

Figure 12 shows a request data graph analyzing the test results of websites accessed without any system, with Ad-Blocker, and using Pi-Hole. It can be concluded that the Ad-Blocker alone is not efficient in reducing the number of requests, as several domains show a total request count that is nearly the same without any system in place. However, when the Pi-Hole DNS Server is implemented, it significantly reduces the number of requests made when clients access a website. In the request data graph for the domain *livescore.in*, the level of effectiveness is high: Without any system, the total number of requests was 476. When using an Ad-Blocker, the requests dropped to 239. After implementing Pi-Hole, the total number of requests further decreased to only 204.

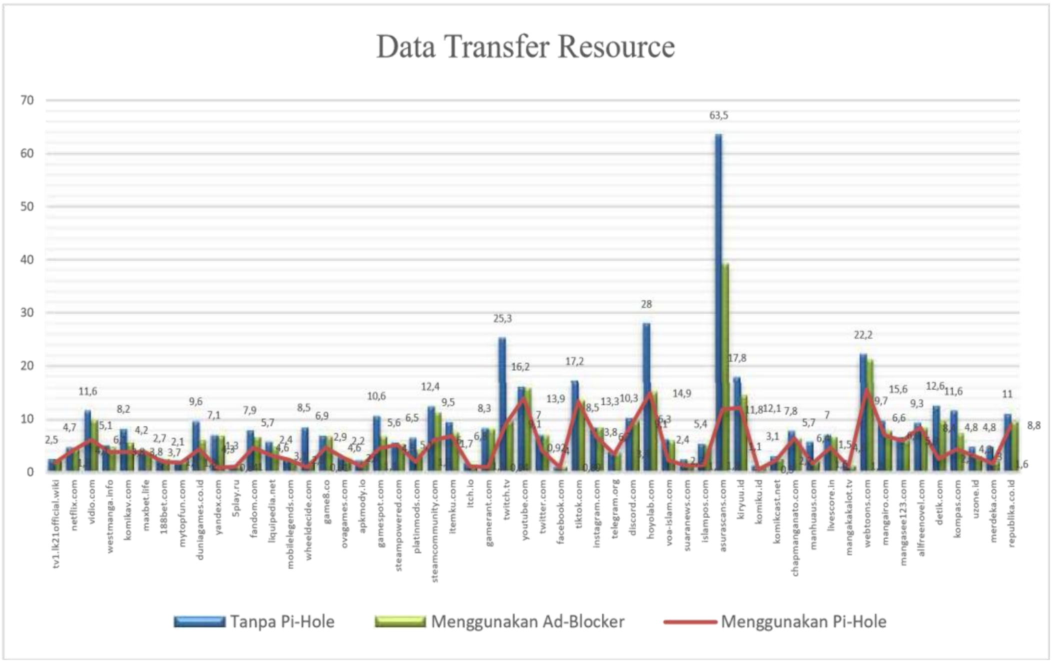


Figure 13. Resource Data Transfer Graph

Figure 13 shows a resource data transfer graph analyzing website testing results without a system, with Ad-Blocker, and using Pi-Hole. It can be concluded that the Ad-Blocker is not efficient in reducing the amount of data transfer resources, as in several domains the total data transfer remains nearly the same with or without the Ad-Blocker. However, when the Pi-Hole DNS Server is applied, it significantly reduces the amount of data transferred when clients access a website. In the data transfer graph for the domain aurascans.com, the level of effectiveness is high: Without any system, the data transfer was 63.5 MB. When using an Ad-Blocker, it was reduced to 39.2 MB. After implementing Pi-Hole, the data transfer dropped significantly to just 11.8 MB.

Figure 14 shows the load time graph analyzing the results of website testing without a system, with Ad-Blocker, and using Pi-Hole. It can be concluded that the Ad-Blocker is not efficient in reducing load time, as for several domains the total load time is almost the same with or without the Ad-Blocker. However, applying the Pi-Hole DNS Server can reduce the load time when clients access a website. In the load time graph for the domain gamerant.com, the effectiveness is notably high: Without any system, the load time was 43.22 seconds. With the Ad-Blocker, the load time dropped to 5.61 seconds. After using Pi-Hole, the load time to load the site further decreased to just 2.74 seconds.

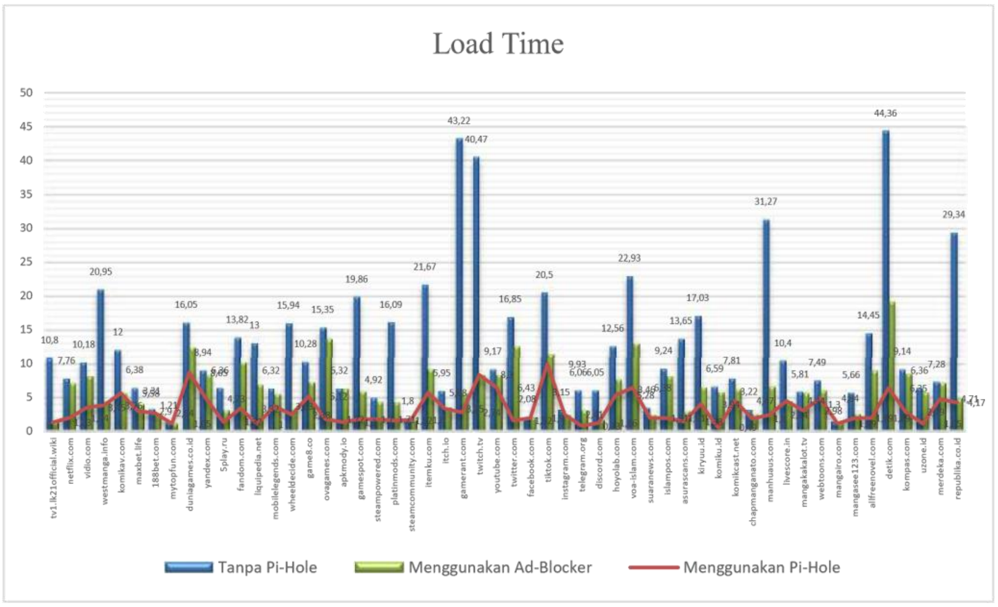


Figure 14. Load Time Graph

The analysis of the ad-blocking test results was conducted to observe the blocking of ads on web pages accessed by clients, including ads on the home page, side page, bottom page, and pop-ups, comparing the scenarios without any system and with the use of Pi-Hole and Ad-Blocker. The domain blocking test analysis was carried out to examine the effectiveness of blocking domains accessed by clients using the Pi-Hole DNS Server that are listed on the blacklist. Below are several domains that have been tested using the Pi-Hole DNS Server.

3.9 *Discussion*

The research results demonstrate the implementation of the Naive Bayes algorithm for domain classification to reduce false positives. This study used domain datasets collected from various sources, including datasets downloaded from the website Kaggle.com. The dataset consisted of 360 web domain entries, 15 pornographic domain entries, and 126 advertising domain entries. The preprocessing process involved several stages such as stop-word removal, case folding, tokenization, and stemming.

The preprocessing process was carried out in two scenarios: one with preprocessing and the other without. The research results showed that data preprocessing does not always improve the performance of the classification model. The scenario with preprocessing yielded lower accuracy rates of 74% and 78%, compared to the scenario without preprocessing, which achieved accuracy rates of 81% and 84%. Based on this data, it was concluded that in the context of domain classification, data preprocessing is not always necessary and may negatively affect classification performance.

The next step before applying the Naive Bayes algorithm was dataset splitting, dividing the data into training and testing sets. The dataset was split randomly with a ratio of 80:20. A total of 400 data entries were used for training the model, and 100 entries were used for testing. Once split, the data underwent feature extraction.

Feature extraction was also conducted under two scenarios: one using the TF-IDF method and the other using the N-gram method. The research results showed that the N-gram method produced better classification performance than TF-IDF. The scenarios using the N-gram method yielded accuracy rates of 78% and 84%, while the scenarios using TF-IDF achieved accuracy rates of 74% and 81%. Based on this data, it was found that selecting the appropriate feature extraction method significantly impacts classification accuracy.

Following feature extraction, the classification process was conducted using the Naive Bayes algorithm. The classification process in this study was carried out using four different scenarios:

1. Scenario 1: TF-IDF feature extraction with data preprocessing.
2. Scenario 2: TF-IDF feature extraction without preprocessing.
3. Scenario 3: N-gram feature extraction with preprocessing.
4. Scenario 4: N-gram feature extraction without preprocessing.

Scenario 1 yielded an accuracy of 74%. Scenario 2 yielded 81%. Scenario 3 achieved 78%, while Scenario 4 achieved the highest accuracy at 84%. These results indicate that the best classification performance was obtained using the N-gram feature extraction method, particularly without preprocessing. In that scenario, classification accuracy reached 84%, while the best TF-IDF scenario without preprocessing yielded 81%. These findings suggest that for classification tasks, the N-gram method may be more suitable than TF-IDF and that preprocessing does not always enhance classification performance. Diverse methods may be needed to further improve performance.

This study demonstrates that the implementation of machine learning methods in domain blocking systems can achieve a relatively high accuracy rate of 84%. This result indicates that the model is capable of effectively distinguishing between malicious and legitimate domains. However, the accuracy is not yet optimal, as there is still a significant rate of false positives and false negatives. Several challenges contribute to the limited accuracy, including the quality and imbalance of the training dataset, the lack of sufficiently representative features, and the suboptimal performance of the machine learning model in recognizing complex patterns associated with malicious domains. Additionally, the absence of contextual information—such as behavioral data or domain access timing—further limits the model's performance.

To improve the quality and accuracy of the system, several strategic steps can be taken. These include expanding and balancing the dataset, developing more contextual features such as domain age, SSL status, and access frequency, and employing more advanced algorithms such as ensemble learning or deep learning techniques. Cross-validation and careful parameter tuning are also essential to ensure the model performs well across different scenarios. Moreover, integrating both static and dynamic analysis can help reduce false positives and enhance the overall reliability of the system. With these efforts, the machine learning-based domain blocking system is expected to achieve higher accuracy and become more effective in detecting and mitigating evolving cyber threats.

The implementation of the Pi-Hole DNS Server as an Ad-Blocker and Website Filtering System in a Computer Network serves to restrict access to negative content unrelated to academic activities on a campus network. Pi-Hole, an open-source DNS server application, is used to block advertisements and undesirable domains, including websites containing harmful content such as pornography, gambling, and dangerous sites. Websites often contain various types of ads, with the most common being paid search, paid social, and video marketing. These online ads can sometimes disrupt user experience.

Once this system is in place, Pi-Hole can block ads and domains on websites. The system works by filtering DNS queries from clients. Pi-Hole matches the queries with blacklist and whitelist entries to determine whether a query should be blocked or forwarded to the client. If the query is on the blacklist, the system blocks it. If it's on the whitelist, it forwards the request to the service provider's DNS server, which then returns the result to the client. Pi-Hole also monitors the activity logs of every device newly connected to the network. The system is implemented using a Mirotic RB951Ui–2HnD router. Before testing, a domain list was created using data from Semrush.com. These domains were tested both with and without the Pi-Hole DNS Server to verify whether the system worked as intended. Several testing scenarios were conducted, including website traffic tests using Pi-Hole and Ad-Blocker, ad-blocking tests, and domain-blocking tests.

Based on traffic testing results, when a client accessed gamespot.com without Pi-Hole and Ad-Blocker, the website took longer to load and displayed ads on the homepage. There were 322 requests, 10.6 MB of resource data transferred, and the total load time was 19.86 seconds. When accessing the same site using the Pi-Hole DNS Server, the total number of requests was reduced to 112, the data transferred was only 4.5 MB, and the load time was significantly faster at just 1.70 seconds, making the site more efficient. For ad-blocking tests, without Pi-Hole and before activating the Ad-Blocker extension, kompas.com displayed many online ads on the homepage, including pop-ups, side page ads, and bottom page ads, which were distracting and slowed down the site. After applying the Pi-Hole DNS Server and activating the Ad-Blocker extension, these ads were blocked and did not disrupt network performance. Users accessing the site experienced a cleaner and faster load time.

In domain-blocking tests without Pi-Hole, a test on youtube.com showed that the site was still accessible and displayed content to the client. After implementing the Pi-Hole DNS Server, youtube.com was successfully blocked and the site could no longer be accessed because it was on the blacklist. From the analysis results, comparing traffic data using Chrome DevTools with and without Pi-Hole and Ad-Blocker showed that total requests, data transfer resources, and load time were all significantly reduced when using Pi-Hole. In the request analysis chart, *livescore.in* had high effectiveness — 476 requests without the system, 239 with Ad-Blocker, and only 204 with Pi-Hole. For data transfer resource analysis, aurascans.com was highly efficient — 63.5 MB without the system, 39.2 MB with Ad-Blocker, and just 11.8 MB with Pi-Hole. In load time analysis, gamerant.com showed high efficiency — 43.22 seconds without the system, 5.61 seconds with Ad-Blocker, and only 2.74 seconds with Pi-Hole.

In ad-blocking test analysis, some whitelisted websites still showed ads on their homepages, which disrupted browsing activities and slowed loading, but these could be blocked using Ad-Blocker with Pi-Hole DNS Server. In the analysis of domain blocking from blacklisted sites, the results showed that the Pi-Hole DNS Server successfully blocked domains that had been listed.

Table 1. Result of Research

Aspect	This research	Previous research
Approach	Machine learning (automatic classification of domains and content)	A manual approach to domain and content classification
Result	Medium to high, depending on training	Low to high, depending on the list entered.
Accuracy	Improved with better datasets and features	Depends on the completeness of the list
Scalability	Applied to high traffic in real-time	Limited – performance degrades if the list is too long

4. Conclusions

Based on the research findings, the results show that better classification performance is achieved using the N-gram feature extraction method, especially in scenarios without data preprocessing. In such scenarios, the classification accuracy reaches 84%, while in the best scenario using the TF-IDF method without preprocessing, the accuracy reaches 81%. These findings indicate that for the classification task, the N-gram method can be a better option than the TF-IDF method, and data preprocessing does not always enhance classification performance.

According to the research results, testing was carried out using several trial scenarios, including website traffic testing without the system using Pi-Hole and Ad-Blocker, ad-blocking tests, and domain-blocking tests. In the website traffic test without the system using Pi-Hole and Ad-Blocker, Chrome DevTools was used to compare website traffic. This test involved observing total requests, data transfer resources, and load time—the total time for all website traffic when accessing the site was higher compared to after using Pi-Hole and Ad-Blocker. In the ad-blocking test without Pi-Hole and Ad-Blocker, ads still appeared on the website and slowed down the page load time. However, after using Pi-Hole and Ad-Blocker, ads no longer appeared on the website and were successfully blocked. In the domain-blocking test, before implementing the Pi-Hole DNS Server, the website pages still displayed information accessed by the client. After implementing the Pi-Hole DNS Server, the website pages became inaccessible as they were included in the blacklist. Based on the research conducted, the implementation of Pi-Hole as a DNS Server in the Computer Lab was successfully carried out. The system is capable of blocking websites and advertisements.

Acknowledgments: This research was made possible with the support of Makassar State University through the provision of valuable facilities and resources. Their support and encouragement played a significant role in the success of this research.

Author contributions: The authors are responsible for building Conceptualization, Methodology, analysis, investigation, data curation, writing—original draft preparation, writing—review and editing, visualization, supervision of project administration, funding acquisition, and have read and agreed to the published version of the manuscript.

Funding: The study was conducted without any financial support from external sources.

Availability of data and Materials: All data are available from the authors.

Conflicts of Interest: The authors declare no conflict of interest.
Additional Information: No Additional Information from the authors.

References

[1] Abdurrahman, O., & dkk (2022). Penerapan PI HOLE DNS Server sebagai ADS - Blocker dan sistem filtering website pada jaringan hotspot. Jurnal Media Infotama, 18(2), 208-217.
[2] Akbar, A. P (2022). Metode block access serta memanejemen bandwith pada Mikrotik Rb951ui dan Mikrotik Rb 941-2ND di Caffe Ready Jombang Jawa Timur. Jurnal Teknologi Dan Sistem Informasi Bisnis, 4(2), 398-406.
[3] Ali, I. S., Hamza, S., Gunawan, E., & Halim, F(2020). Implementasi & analisis penerapan pi-hole network ad-blocking di Laboratorium Jaringan Teknik Informatika UMMU. Jurnal Teknologi Informatika, 27-31.
[4] Alvaro Feal (2021) . Blocklist babel: On the transparency and dynamics of open source blocklisting. IEEE Transactions on Network and Service Management. doi 10.1109/TNSM.2021.3075552,
[5] Apriyatna, M., & Zulfikar, A. F (2023). Analisis dan implementasi network adblocking pi-hole di Raspberry Pi 4 menggunakan OPNSense DHCP dengan metode PPDIOO (Studi kasus DiskominfoSP Kabupaten Lebak). Jurnal Ilmu Komputer dan Science, 2(2), 575-582.
[6] Chazar, C., & Widhiaputra, B. E (2020). Machine learning diagnosis kanker payudara menggunakan algoritma support vector machine. Informasi (Jurnal Informatika dan Sistem Informasi), 12(1), 67-80.

[7] Farmadiansyah, A. Z (2021). Deteksi surel spam dan non-spam bahasa indonesia menggunakan metode naive bayes. Skripsi.Yogyakarta: Universitas Islam Indonesia.

[8] Feal, A., Vallina, P., Gamba, J., Pastrana, S., Nappa, A., Hohlfeld, O., Vallina-Rodriguez, N., & Tapiador, J. (2021). Blocklist Babel: On the Transparency and Dynamics of Open Source Blocklisting. IEEE Transactions on Network and Service Management, 18(2), 1334–1349. https://doi.org/10.1109/TNSM.2021.3075552

[9] Husen, Z., & Surbakti, M. S (2020). Membangun server dan jaringan komputer dengan Linux Ubuntu. Skripsi. Aceh: Syiah Kuala University.

[10] Inayah, K., & Ramli, K. (2024). Analisis Kinerja Intrusion Detection System Berbasis Algoritma Random Forest Menggunakan Dataset Unbalanced Honeynet BSSN. Jurnal Teknologi Informasi Dan Ilmu Komputer, 11(4), 867–876. https://doi.org/10.25126/jtiik.1148911

[11] Mujiastuti, R., & Prasetyo, I (2021). Membangun sistem keamanan jaringan berbasis VPN yang terintegrasi dengan DNS Filtering PIHOLE. Jurnal Universitas Muhammadiyah Jakarta, 1-10.

[12] Mulyana, D. I., Ardiyansyah, F., Hidayat, N., & Zulfikar, A. (2024). Optimasi Keamanan Jaringan Wifi dari Situs Judi Online dan Pornografi dengan DNS Filtering dan Orangepi. MALCOM: Indonesian Journal of Machine Learning and Computer Science, 4(2), 647–655. https://doi.org/10.57152/malcom.v4i2.1274

[13] Munawar, Z., & Putri, N. I (2020). Keamanan jaringan komputer pada era big data. J-SIKA: Jurnal Sistem Informasi Karya Anak Bangsa, 2(01), 14-20.

[14] Nursiyono, J. A., & Huda, Q (2023). Analisis sentimen twitter terhadap perlindungan data pribadi dengan pendekatan machine learning. Jurnal Pertahanan dan Bela Negara, 13(1), 1-16.

[15] Prasetya, Hanif, H., & Handaga, B (2023). Implementasi pemanfaatan Pi-Hole sebagai DNS Server Pada Rumah untuk memonitoring traffic internet dan memblokir iklan. Jurnal Universitas Muhammadiyah Surakarta, 1-16.

[16] Rahman, M (2023). Implementasi web content filtering pada jaringan RT/RW Net menggunakan Pi-Hole DNS Server. Jurnal Generation Journal, 50-60.

[17] Satriawan, D., & Trisnawan, P. H (2021). Implementasi layanan dns sinkhole sebagai pemblokir iklan menggunakan arsitektur cloud. Jurnal Informatika, 67-75.

[18] Sidik, F., Maryati, M., & Abdullah, A. (2023). Implementasi Dns (Domain Name System) Adblocker Menggunakan Raspberry Pi 4 Pada Politeknik Piksi Input Serang. Jurnal Gerbang STMIK Bani Saleh, 13(1),60–73.

[19] Studi, P., Multimedia, T., Jaringan, D. A. N., Teknik, J., Dan, I., & Jakarta, P. N. (2023). Program studi teknik multimedia dan jaringan jurusan teknik informatika dan komputer politeknik negeri jakarta 2023

[20] Suryanto, & Permadi, F. A (2019). Optimalisasi internet hotspot menggunakan user manajemen pada Pusat Pengembangan SDM Asuransi Indonesia. Jurnal Infortech, 1, 60-61.

[21] W, Y., Fitriana, Y. B., Susanto, A., Susanto, E. S., Hamdani, F., Rizky, M., & Oper, N (2022). Implemetasi filtering alamat website pada web proxy menggunakan Raspberry-Pi. Jurnal Pengembangan IT, 55-61.

[22] Wibawa, A. P., Purnama, M. G., Akbar, M. F., & Dwiyanto, F. A. 2018. Metode-metode klasifikasi. Prosiding Seminar Ilmu Komputer dan Teknologi Informasi. 3, pp. 134-138.

[23] Widiatmoko, C (2022). Rancang bangun secure mobile router dan sistem pemblokiran konten iklan dan kostumisasi domain berbasis raspberry pi 3b+. Disertasi. Jakarta. Politeknik Negeri Jakarta.

[24] Wuhi, A. U., Hariadi, F., & Uly, N. B. (2024). Implementasi Web Filtering Firewall Untuk Mendukung Internet Sehat Di Smp Negeri 4 Mauliru (Implementation Of Web Filtering Firewall To Support a Healthy Internet At SMP Negeri 4 Mauliru). 3(1), 43–52.

[25] Zakariah, M. A., Afriani, V., & Zakariah, K. M (2020). Metodologi penelitian kualitatif, kuantitatif, action research, research and development (R n D). Jurnal Penelitian. Kolaka: Yayasan Pondok Pesantren Al Mawaddah Warrahmah.