International Conference on Interdisciplinary Studies in Education, Research and Technology 2024 (ISERT-2024)
Available at www.ijsred.com

RESEARCH ARTICLE OPEN ACCESS

Analyzing URL Structure for Machine Learning: Feature Engineering and Classification Applications

Venkata Sai Chandradeep Telaprolu
chandradeep.msba@gmail.com

***Abstract*—The Uniform Resource Locator (URL) is a fundamental element of internet traffic that provides critical information for a variety of machine learning applications, including security threat detection and content classification. Despite its ubiquity, extracting meaningful features from URLs remains a complex task due to their structural variability and context-dependent nature. This paper presents an in-depth exploration of URL feature engineering techniques. I define key URL components, such as domain, path, and protocol, and outline methods for generating features that effectively capture the inherent structure of these components. Additionally, I introduce an open-source Python package, url2features, designed to automate the feature extraction process and integrate seamlessly with machine learning workflows. Using a variety of URL-based classification tasks, I demonstrate the impact of these features on model performance and analyze feature importance across different datasets. My results highlight the value of structured URL feature engineering for enhancing the interpretability and effectiveness of machine learning models in diverse domains such as cybersecurity and content categorization.**

I. Introduction

Uniform Resource Locators (URLs) play a critical role in the digital landscape, serving as key mechanisms for accessing information and connecting users with content. URLs facilitate access to news, media, advertisements, businesses, and social platforms, while also being heavily utilized by online services in background operations for communication and data exchange. This duality emphasizes the importance of URLs for both human users and automated systems.

URL processing is crucial for numerous internet data analysis tasks. In many real-time applications, URLs often represent the primary piece of information due to the need for rapid examination. For example, in detecting malicious websites, URLs must be analyzed quickly and efficiently to maintain effectiveness [1]. Common scenarios for URL-focused analysis include evaluating potential phishing attempts [2], [3], [4], [5], [6], [7], [8], [9], [10] and identifying sites that may distribute malware or other harmful software [11], [5].

URL analysis also extends to areas like online advertising, where tasks include predicting conversions [12] and performing contextual analysis of webpage content [13], [14], [15], [16], [17], [18], evaluating webpage relevance [19], and analyzing language [20]. In addition, categorization efforts based solely on URLs have proven effective in filtering spam webpages from search results [21], [22].

At its essence, a URL consists of several components, with the domain being a critical element. The domain often reveals insights into the intended use, authenticity, and potential country of origin of a URL. Furthermore, querying Domain Name Servers (DNS) allows for additional information gathering, including the domain's history and its network topology.

Beyond domains, a URL's structure comprises sequences of words, categories, identifiers, dates, or domain-specific abbreviations. These components can hint at how information is organized or the internal logic of applications that dynamically generate or display content.

Feature engineering methods for URLs often utilize string patterns and regular expressions to detect key sequences and lexical attributes [13], [2], [5], [8]. Another approach involves using lookup tables and scoring based on key URL sequences [16]. Strategies such as n-gram and bag-of-words models are also employed [15], [6], as well as task-specific word embeddings derived from URL segmentation [23], [12]. Additional context can be provided through domain name servers and registrar data, offering insights on domain registration and server configuration [11], [10].

Modern malicious URL detection methods often combine several feature engineering techniques [24], [10]. More sophisticated approaches integrate, select, or dynamically learn from different features, adapting to evolving challenges, especially within cybersecurity [25], [10].

Some researchers emphasize treating URLs as sequences of characters, necessitating sequentially aware feature extraction methods [23], [7]. This has led to the development of neural network models utilizing convolutional or recurrent layers to capture sequential URL patterns. However, unlike human language structures with flexible grammar, URLs maintain a consistent order of key elements (shown in Figure 1). This predictable structure can be leveraged for effective algorithm design [14]. Furthermore, psychological and social factors in URL design, like the use of popular brand names in subdomains to mislead users in phishing attempts [8], further highlight the value of structure-sensitive feature engineering.

In general, feature engineering strategies are tailored to specific tasks, but systematic evaluations across multiple applications remain limited.

In this work, I introduce a feature engineering library based on a URL ontology, enabling structured feature generation and visualization of feature importance. These tools are especially useful for tasks requiring adaptation to shifting challenges, with diverse feature needs and efficient selection [3]. I provide an open-source toolkit for generating these structured features and demonstrate their effectiveness across various tasks, delivering a cross-domain evaluation of URL feature

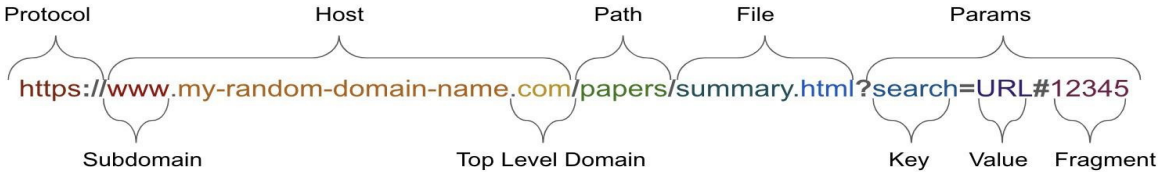

Fig. 1. Structural elements of a URL that necessitate specialized feature engineering techniques

TABLE I
URL COMPONENTS AND RELATED FEATURE DATA

Component	Subcomponent	Type of Feature Data
Protocol		Indicates content type, transmission technology, security level
Domain		Textual data reflecting overall intent
	Subdomain	Specific function or categorization of content
	Top-Level Domain	Purpose of business/service, authenticity, geographic origin, age
Path		Describes content organization, file system, application structure
File		Represents naming conventions, psychological intent, file types (static/executable)
Parameters		Sub-content structure, personalization details, security elements, tracking information
	Keys	Categories distinguishing sub-content organization
	Values	Values that describe sub-content, indicate page elements
	Fragment	Labels for sub-structure or points of customization

utility. My methodology enhances interpretability in machine learning model decisions, contrasting with the ”black-box” approach typical of neural networks that treat URLs merely as text sequences. Additionally, the cross-task analysis offers valuable insights into what strategies succeed across different domains, crucial for applications that must rapidly detect new fraud or behavioral patterns from limited datasets.

II. METHODOLOGY

The structural makeup of a URL (as illustrated in Figure 1) is detailed in Table I. This table provides an overview of the information inherent in each URL component and describes how these components contribute to creating features for machine learning models.

We developed a library for extracting features from URLs, generating component-specific features through an organized naming convention that allows for separation and clustering during analysis. Prior research has provided datasets structured by features for particular problems [26]. In contrast, my application enables extraction and evaluation of any desired subset of these features for various tasks.

We applied this library to diverse machine learning tasks using URLs as the primary data input. The tasks were selected to cover a broad spectrum of problem types and data sizes. Standard machine learning models were employed to assess the impact of the extracted URL features, with feature importance evaluated through the SHAP package. This enabled detailed analysis of features within the URL's logical structure.

A. Data

The datasets used in this study were compiled from multiple sources, providing a diverse set of URL-based classification challenges. The ISCX Malicious URL dataset distinguishes between benign and malicious URLs such as those used in spam, malware, and phishing attacks [5]. The WebKb 4 Universities dataset covers a variety of university-related URLs categorized by topic [27]. The Syskill & Webert dataset contains human-rated webpages in four categories for content preference prediction [28]. Additionally, I utilized the Kaggle DMOZ dataset, which features a large number of topic- categorized URLs [29].

TABLE II
DATASETS OVERVIEW

Dataset	Records	Classes	Largest Class Proportion
spam	47,378	2	75%
Malware	46,944	2	75%
Phishing	45,343	2	78%
WebKb 4Uni	8,284	7	45%
Syskill & Webert	330	3	68%
DMOZ	1,562,978	15	16%

A summary of these six URL-based classification tasks is shown in Table II, which details the number of records, class counts, and the percentage of the largest class.

For each dataset, I generated comprehensive URL features and employed various machine learning models to build classifiers. These were analyzed to determine the key features relevant to each task.

B. Feature Generation

We implement a feature extraction process that focuses on different segments of a URL, allowing users to select specific feature sets relevant to their needs. In addition, I generate global features that capture overall characteristics of the URL string as a whole, rather than isolating individual sections. The full list of features, along with their categorization and descriptions, is summarized in Table III.

Our URL feature extraction functions are available through an open-source Python package, *url2features*. This tool is designed to process large datasets in chunks, only generating and adding feature groups as specified by the user. The naming system ensures that features tied to specific URL regions can be identified and analyzed collectively.

```
pip install url2features
url2features -columns=URLCOL -host -tld input.csv >
    output.csv
```

Listing 1. Installation and Usage of url2features via Command Line

TABLE III
URL FEATURES

Group	Feature	Type	Description
Protocol	protocol_name	Category	Specifies the internet protocol used
	protocol_type	Category	Classifies the protocol by its primary function
	protocol_exists	Boolean	Indicator for the presence of a protocol
Host	host_is_ip	Boolean	Indicates if the host is an IP address
	host_has_port	Boolean	Identifies if a port number is defined
	domain_len	Numeric	Number of characters in the domain (excluding TLD and subdomains)
	domain_alpha	Numeric	Proportion of alphabetic characters in the domain name
	domain_sections	Numeric	Number of segments in the domain, separated by periods
	subdomain	Category	Value representing the subdomain
	subdomain_type	Category	Categorizes the subdomain's function or intent
	subdomain_freq	Numeric	Usage frequency of the subdomain in network traffic
	tld_name	Category	Name of the top-level domain (TLD)
	tld_type	Category	Categorizes the TLD by purpose
	tld_freq	Numeric	Occurrence frequency of the TLD in network traffic
Path	path_depth	Numeric	Level of directory hierarchy between host and file
	path_1st_wd	Category	First distinct word in the path with more than 2 characters
	path_1st_wd_prefix	Category	First 3 characters of the initial word in the path with more than 2 characters
	path_wd_count	Numeric	Count of distinct words in the path
	path_wd_len	Numeric	Average length of words in the path
	path_has_date	Boolean	Flag for presence of a date in the path
	path_is_home	Boolean	Indicator for home directory path
File	file_len	Numeric	Length of the target file name in characters
	file_1st_wd	Category	Initial distinct word in the file name
	file_1st_wd_prefix	Category	First 3 characters of the initial distinct word in the filename
	file_wd_count	Numeric	Number of words over 2 characters in the file name
	file_wd_len	Numeric	Average length of words in the file name
	file_ext	Category	File extension, such as "exe" or "php"
	file_type	Category	Categorization of the file extension by its function (e.g., static or dynamic)
	file_ext_exists	Boolean	Indicator for the presence of a file extension
Params	params_len	Numeric	Character count of the parameter string
	params_count	Numeric	Number of key-value pairs in the parameter string
	params_match	Boolean	Indicator for matched key-value count in parameters
	params_has_url	Boolean	Flag for raw URL presence within the parameters
	params_enc_url	Boolean	Flag for encoded URL presence in the parameters
	params_enc_char	Boolean	Flag for encoded character presence in the parameters
	params_frag_len	Numeric	Length of the fragment part of the parameter string
	keys_count	Numeric	Count of unique keys within the parameter string
	keys_len	Numeric	Average length of keys in the parameter string
	keys_numeric	Numeric	Proportion of numeric characters in keys
	values_count	Numeric	Number of unique values in the parameter string
	values_len	Numeric	Average length of values in the parameter string
	values_numeric	Numeric	Proportion of numeric characters in values
	frag_len	Numeric	Character length of the fragment string
	frag_secs	Numeric	Number of sections in the fragment string separated by delimiters [?&=]
	frag_enc_char	Boolean	Flag for encoded character presence in the fragment string

Listing 1 displays the command-line instructions for installing and using the *url2features* package. Running this command applies feature extraction to the *URLCOL* column within the *input.csv* file. Features created are specified by the `-host` and `-tld` switches, which extract characteristics related to the host and top-level domain. By default, output is directed to STDOUT, so here the output is redirected to *output.csv*.

```
from url2features.pipeline import URLTransform
from sklearn.linear_model import SGDClassifier
from sklearn.pipeline import Pipeline

pipeline = Pipeline([
    ('url_transform', URLTransform(['URLCOL'], ['
    host', 'tld'])),
    ('classifier', SGDClassifier(loss='log')),
])
```

Listing 2. Utilizing url2features in Scikit-Learn Pipelines

The package can also be incorporated programmatically into machine learning workflows, as illustrated in Listing 2. Here, the `host` and `tld` features are transformed within a scikit-learn pipeline, demonstrating seamless integration for feature engineering within a serialized machine learning model. This setup simplifies deployment and enhances the modularity of transformations.

C. Assessing Feature Relevance

To assess the importance of features, I employ the SHAP library [30] on a small test dataset comprising 80-100 samples per experiment. By aggregating Shapley values, I derive the average absolute impact of each feature, akin to standard visualizations of Shapley feature importance.

In my analysis of URL-based features, I group feature importance values according to the URL segments from which they originate, following the structure depicted in Figure 1. Summing feature importance for segments allows visualizing their cumulative contributions to model performance. I also plot a faint grey line to indicate global feature relevance, such as the overall URL length. If non-URL-specific features are present, they are aggregated into the global grouping.

This visualization illustrates how different URL components influence model predictions, facilitating the comparison of feature importance across the six study tasks. A script for creating such visualizations is included in the *url2features* package source code.

III. PERFORMANCE EVALUATION

Machine learning pipelines were tested on each of the six datasets using default settings without parameter optimization. Feature preprocessing was tailored to suit each model class, with comprehensive details provided in the experimental source repository. Table IV summarizes model performance using Balanced Accuracy, which is a suitable metric across all six tasks.

Table IV demonstrates that security-related tasks perform exceptionally well with URL-based classification models. Conversely, topic and user-preference tasks exhibit relatively modest performance, likely due to greater target variability and smaller datasets in two out of three cases. However, even

TABLE IV
PERFORMANCE OF MACHINE LEARNING MODELS

Dataset	NB	LR	XT	LGBM
Spam	1.00	1.00	1.00	1.00
Malware	0.65	0.93	0.99	1.00
Phishing	0.89	0.98	0.99	1.00
WebKb 4Uni	0.30	0.36	0.58	0.49
Syskill & Webert	0.39	0.35	0.37	0.33
DMOZ	0.12	0.17	0.29	0.25

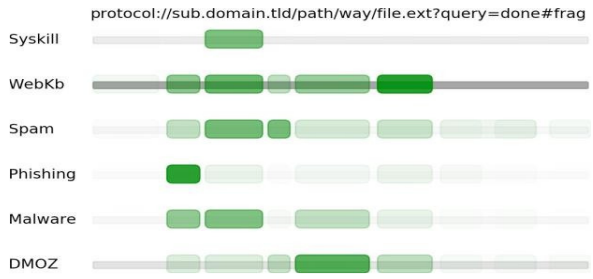

Fig. 2. Structural Depiction of URL Feature Importance

with the large DMOZ dataset, URL-based topic classification appears inherently challenging.

To highlight feature importance, I used the top-performing model for each task and generated SHAP values for test samples. The results are aggregated by the URL segments from which features are derived, as depicted in Figure 2. This approach provides insights into the signal sources for each task, enabling cross-comparison of feature contributions.

Results reveal that signal strength varies significantly across tasks. While protocol and parameter features provide minimal signal in most cases, domain, path, and file components are more influential. Specific differences also emerge, such as the importance of subdomains for phishing detection and the influence of domain-to-file path features in WebKb classification. Security-related problems rely less on global features compared to topic classification tasks, as indicated by the prominence of grey lines representing global feature contributions. This suggests that security problems derive more predictive power from granular URL components.

Overall, feature contributions vary considerably based on task requirements, implying that data scientists should rigorously evaluate features rather than depending solely on general heuristics.

IV. CONCLUSION

This study has outlined the core structural elements of a URL from which specific contextual features can be derived. These features are typically tailored to distinct problem types, with adjustments made to serve particular analysis needs. In this paper, I introduced an open-source package for generating URL-based features, designed to be both aware of the information source within each URL component and adaptable for a wide range of applications. I applied these features in experiments across diverse URL classification tasks.

Our findings reveal that various structural segments of a URL contribute differently across tasks, with distinct levels of significance. Certain trends, such as the impact of global URL characteristics, seem to relate broadly to task categories, while others show unique, task-specific relevance. Notably, the domain name and its subcomponents consistently enhance model performance across tasks, though the specific insights drawn from the domain vary. Nevertheless, the domain remains a valuable and potent source of information for machine learning models that utilize URLs as a primary feature.

REFERENCES

[1] J. Ma, L. K. Saul, S. Savage, and G. M. Voelker, "Identifying suspicious urls: An application of large-scale online learning," in *Proceedings of the 26th Annual International Conference on Machine Learning*, ser. ICML '09. New York, NY, USA: Association for Computing Machinery, 2009, p. 681–688. [Online]. Available: https://doi.org/10.1145/1553374.1553462

[2] S. Garera, N. Provos, M. Chew, and A. D. Rubin, "A framework for detection and measurement of phishing attacks," in *WORM'07*, Alexandria, VA, 2007. [Online]. Available: http://doi.acm.org/10.1145/1314389.1314391

[3] R. B. Basnet and Q. Sung, Andrew H.and Liu, "Feature selection for improved phishing detection," in *Advanced Research in Applied Artificial Intelligence*, H. Jiang, W. Ding, M. Ali, and X. Wu, Eds., vol. 7345. Berlin, Heidelberg: Springer Berlin Heidelberg, 06 2012, pp. 252–261.

[4] R. Basnet, "Learning to detect phishing urls," *International Journal of Research in Engineering and Technology*, vol. 03, pp. 11–24, 06 2014.

[5] M. S. I. Mamun, M. A. Rathore, A. H. Lashkari, N. Stakhanova, and A. A. Ghorbani, "Detecting malicious urls using lexical analysis," in *Network and System Security*, J. Chen, V. Piuri, C. Su, and M. Yung, Eds. Cham: Springer International Publishing, 2016, pp. 467–482.

[6] R. Verma and A. Das, "What's in a url: Fast feature extraction and malicious url detection," in *Proceedings of the 3rd ACM on International Workshop on Security And Privacy Analytics*, ser. IWSPA '17. New York, NY, USA: Association for Computing Machinery, 2017, p. 55–63. [Online]. Available: https://doi.org/10.1145/3041008.3041016

[7] A. Vazhayil, V. Ravi, and S. Kp, "Comparative study of the detection of malicious urls using shallow and deep networks," in *Unknown*, 07 2018, pp. 1–6.

[8] H. Tupsamudre, A. Singh, and S. Lodha, *Everything Is in the Name – A URL Based Approach for Phishing Detection.* Unknown, 05 2019, pp. 231–248.

[9] S. S. Sirigineedi, J. Soni, and H. Upadhyay, "Learning-based models to detect runtime phishing activities using urls," in *Proceedings of the 2020 the 4th International Conference on Compute and Data Analysis*, ser. ICCDA 2020. New York, NY, USA: Association for Computing Machinery, 2020, p. 102–106. [Online]. Available: https://doi.org/10.1145/3388142.3388170

[10] T. Li, G. Kou, and Y. Peng, "Improving malicious urls detection via feature engineering: Linear and nonlinear space transformation methods," *Information Systems*, vol. 91, p. 101494, 07 2020.

[11] D. Canali, M. Cova, G. Vigna, and C. Kruegel, "Prophiler: A fast filter for the large-scale detection of malicious web pages," in *Proceedings of the 20th International Conference on World Wide Web*, ser. WWW '11. New York, NY, USA: Association for Computing Machinery, 03 2011, p. 197–206. [Online]. Available: https://doi.org/10.1145/1963405.1963436

[12] Y. Qiu, N. Tziortziotis, M. Hue, and M. Vazirgiannis, "Predicting conversions in display advertising based on url embeddings," in *Proceedings of AdKDD 2020*, 08 2020.

[13] M.-Y. Kan, "Web page classification without the web page," in *Proceedings of the 13th International World Wide Web Conference on Alternate Track Papers & Posters*, ser. WWW Alt. '04. New York, NY, USA: Association for Computing Machinery, 2004, p. 262–263. [Online]. Available: https://doi.org/10.1145/1013367.1013426

[14] L. K. Shih and D. R. Karger, "Using urls and table layout for web classification tasks," in *Proceedings of the 13th International Conference on World Wide Web*, ser. WWW '04. New York, NY, USA: Association for Computing Machinery, 2004, p. 193–202. [Online]. Available: https://doi.org/10.1145/988672.988699

[15] E. Baykan, M. Henzinger, L. Marian, and I. Weber, "Purely url-based topic classification," in *WWW'09 - Proceedings of the 18th International World Wide Web Conference*, 01 2009, pp. 1109–1110.

[16] Sara-Meshkizadeh and A. Masoud-Rahmani, "Webpage classification based on compound of using html features & url features and features of sibling pages," *International Journal of Advancements in Computing Technology*, vol. 2, no. 4, oct 2010.

[17] I. Herna´ndez, C. Rivero, D. Ruiz, and J. L. Arjona, "An experiment to test url features for web page classification," *Advances in Intelligent and Soft Computing*, vol. 157, pp. 109–116, 01 2012.

[18] C. Arya and S. K. Dwivedi, "News web page classification using url content and structure attributes," in *2016 2nd International Conference on Next Generation Computing Technologies (NGCT)*, 2016, pp. 317–322.

[19] M.-Y. Kan and H. O. N. Thi, "Fast webpage classification using url features," in *Proceedings of the 14th ACM International Conference on Information and Knowledge Management*, ser. CIKM '05. New York, NY, USA: Association for Computing Machinery, 01 2005, p. 325–326. [Online]. Available: https://doi.org/10.1145/1099554.1099649

[20] E. Baykan, M. Henzinger, and I. Weber, "A comprehensive study of techniques for url-based web page language classification," *ACM Trans. Web*, vol. 7, no. 1, mar 2013. [Online]. Available: https://doi.org/10.1145/2435215.2435218

[21] Y.-j. Chung, M. Toyoda, and M. Kitsuregawa, "Topic classification of spam host based on urls," in *DEIM Forum 2010 B4-3*, 09 2010.

[22] N. Hassan and I. Hmeidi, "Web spam detection using machine learning specific domain features," *Journal of Information Assurance and Security*, vol. 3, pp. 220–229, 09 2008.

[23] H. Le, Q. Pham, D. Sahoo, and S. C. H. Hoi, "Urlnet: Learning a url representation with deep learning for malicious url detection," *ArXiv*, vol. abs/1802.03162, 02 2018.

[24] A. M. D. Anjali B. Sayamber, "On url classification," *International Journal of Computer Trends and Technology (IJCTT)*, vol. 12, no. 5, p. 235, jun 2014. [Online]. Available: http://www.ijcttjournal.org

[25] S. Sountharrajan, M. Nivashini, S. K. Shandilya, E. Suganya, A. Bazila Banu, and M. Karthiga, *Dynamic Recognition of Phishing URLs Using Deep Learning Techniques*. Cham: Springer International Publishing, 2020, pp. 27–56. [Online]. Available: https://doi.org/10.1007/978-3-030-19353-9_3

[26] G. Vrbanc˘ic˘, I. Fister, and V. Podgorelec, "Datasets for phishing websites detection," *Data in Brief*, vol. 33, p. 106438, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2352340920313202

[27] M. Craven, D. DiPasquo, D. Freitag, A. McCallum, T. Mitchell, K. Nigam, and S. Slattery, "Learning to extract symbolic knowledge from the world wide web," in *Proceedings of the 15th National Conference on Artificial Intelligence (AAAI-98)*. The AAAI Press, 1998.

[28] M. Pazzani, J. Muramatsu, and D. Billsus, "Syskill & webert: Identifying interesting web sites," in *Proceedings of the Thirteenth National Conference on Artificial Intelligence - Volume 1*, ser. AAAI'96. AAAI Press, 1996, p. 54–61.

[29] Kaggle, "Url classification dataset [dmoz]," 2018. [Online]. Available: https://www.kaggle.com/datasets/shawon10/url-classification-dataset-dmoz

[30] S. M. Lundberg and S.-I. Lee, "A unified approach to interpreting model predictions," in *Advances in Neural Information Processing Systems 30*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds. Curran Associates, Inc., 2017, pp. 4765–4774. [Online]. Available: http://papers.nips.cc/paper/7062-a-unified-approach-to-interpreting-model-predictions.pdf

