What's in a Domain? Analysis of URL Features

John Hawkins

Transitional AI Research Group
Getting-Data-Science-Done.com

Abstract

Many data science problems require processing log data derived from web pages, APIs or other internet traffic sources. URLs are one of the few ubiquitous data fields that describe internet activity, hence they require effective processing for a wide variety of machine learning applications. While URLs are structurally rich, the structure can be both domain specific and subject to change over time, making feature engineering for URLs an ongoing challenge.

In this research we outline the key structural components of URLs and discuss the information available within each. We describe methods for generating features on these URL components and share an open source implementation of these ideas. In addition, we describe a method for exploring URL feature importance that allows for comparison and analysis of the information available inside URLs. We experiment with a collection of URL classification datasets and demonstrate the utility of these tools. Package and source code is open on https://pypi.org/project/url2features

Keywords

Machine Learning, Feature Engineering, Web Search, Semantic Web, Data Science

1. Introduction

The Uniform Resource Locator (URL) is a ubiquitous element in the digital world. We use them to retrieve news and media, advertise or find businesses and services, and to interact with other people across a broad range of social applications. Below the surface many online services use URLs internally for communicating with other digital services, making the URL a fundamental data point for both users and machines.

Processing URLs is a key component of many tasks that involve analysing internet data. In many applications the URL is the central piece of information available because the demands of the task require immediate analysis. In applications like malicious wesite detection the URL needs to be processed in a rapid and efficient manner to provide utility[1]. Common use cases for URL centered analysis include security analysis of potential phishing attacks[2, 3, 4, 5, 6, 7, 8, 9, 10], and identification of pages that host malware or viruses [11, 5]. There are also applications to online advertising, including anticipation of conversions [12], contextual analysis of the content[13, 14, 15, 16, 17, 18], relevance [19] or language[20] of webpages. Content categorisation has also been applied to spam web pages using URLs alone for the sake of search results filtering[21, 22].

The URL is made up of multiple elements, but at its core is the domain. The domain contains internal information about both the intended purpose of a URL, its legitimacy and the likely country of origin. The domain is also a source of additional information through requests to Domain Name Servers (DNS) to understand both its history and the structure of its network topology and resources.

Beyond the domain, the URL consists of a sequence of words, categories, identifiers, dates

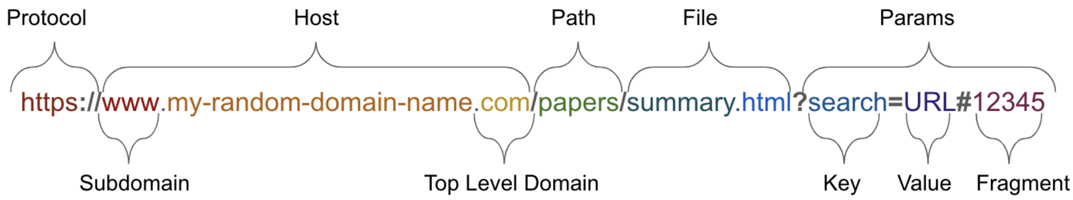

Figure 1: Structural components of a URL requiring specific feature engineering treatment

or domain specific abreviations. This sequence can indicate the psychological elements of how information is categeoried, or the internal logic of an application that generates or presents content dynamically.

Common approaches to feature engineering on URLs include string patterns and regular expressions to identify key sequences and lexical properties [13, 2, 5, 8], creation of lookup tables or likelihood scores based on key sequences [16], n-gram or bag of words models [15, 6], the generation of task specific word embedding vectors based on a segmentation of the URL [23, 12] and the usage of domain name servers or registrars for ancillary information about the domain registration and server configuration[11, 10]. In many modern approaches to malicious URL detection multiple feature engineering approaches are typically combined[24, 10]. Increasingly, sophisticated methods are used to combine, select or learn from combinations of features to adapt to changing requirements, particularly in internet security applications[25, 10].

Some authors have emphasized that URLs are sequences of characters and require feature extraction methods that respect this sequential nature[23, 7]. They have turned to developing neural network approaches based on convolutional or recurrent layers for learning these sequential structures. However, the ordering of key features within URLs is predominantly fixed as shown in Figure 1. The core elements of URL structure occur at fixed positions within the sequence, in contrast to human language sequences which are built from flexible grammars that allow variable positioning of most key elements. The inherent structure of URLs has been demonstrated to be a source of information that can be exploited in algorithm design [14]. Furthermore, there are psychological and social elements to URL construction (such as the use of common brand names in subdomains for phishing attack URLs [8]), which emphasize the utility of feature engineering techniques that respect the inherent structure of URLs.

Typically, researchers employ feature engineering strategies to exploit these potential sources of information in ways that are appropriate for a given task. However, while there are wide variety of approaches that are inconsistently applied, there have been limited studies into the overall effectiveness of different strategies across a range of applications.

In this work we develop a feature engineering library that maintains an ontology of URL features related to underlying structure of the URL. In addition we provide a method for visualising the importance of features across the URL structure. These techniques can prove useful for analysing features for problems that consist of moving targets and require broad sets of potential features and efficient feature selection [3]. We present an open source package for generating features within these ontologocal classes and apply them to a variety of tasks. We present our results as a cross-task evaluation of URL features. The strength of our approach lies in the ability to both understand how machine learning models use URLs in decision making, as opposed to the black-box problem with

Table 1: URL Components and Feature Information

Component	Subcomponent	Type of Feature Information
Protocol		Type of content, technology, security of transmission
Domain		Textual data indicating general purpose
	Subdomain	Specific purpose or categorisation of content
	Top Level Domain	Business or service purpose, authenticty, geograpahy, age
Path		Content structure, file or operation system, application type or structure
File		Naming of specific content, psychological intent, executable or static types
Parameters		Substructure of content, personalization, security, tracking
	Keys	Set of categories used to differentiate substructure
	Values	Set of values for substructure, indication of page content
	Fragment	Labels for specifc substructure, source of new application customisation

neural network models that treat the URL as a text sequence. In addition, the cross-task evaluation we perform provides insight into what works within different domains. This is critical as many machine learning problems that rely on URLs require models built on relatively small datasets, so that new patterns of fraud and behaviour can be detected rapidly.

2. Methodology

The structural components of a URL (as illustrated in Figure 1) are listed in Table 1. We include an overview of the information source in each of those components and highlight how these components relate to machine learning features.

We design a feature extraction library for URLs that generates features specific to each component of the URL independently using a naming convention that permits separation and grouping for analysis. Previous work has presented datasets for specific problems where the features are provided with a similar structural grouping[26]. However, we provide our feature extraction application designed so that any specific subset of these features can be extracted and analysed for a specific task.

We apply this library to multiple machine learning tasks using only URLs as the input variable. We choose these tasks such that they span very different classes of problems and dataset sizes. We experiment with multiple standard machine learning techniques and evaluate the impact of the URL features using the SHAP package for feature importance. We develop a process for analysing these features within the logical structure of the URL.

2.1. Data

The data used in this study was colated from a range of sources to represent a variety of Internet resource classification problems containing URLs as the primary feature. The ISCX Malicious URL classification dataset contains malware, spam and phishing URLS that need to be discriminated against a set of benign URLs[5]. The world wide web knowledge base (WebKb) 4 Universities data set contains a wide range of university URLs categorised into multiple topic categories[27]. The Syskill & Webert webpage ratings dataset containing webepages across 4 categories with a human generated categorisation for determining personal preferences in webpage content[28]. Finally, we utilise the Kaggle DMOZ dataset that contains a large set of topic categorised URLs[29].

These 6 URL classification problems are summarised in Table 2, where we show the number of records, the cardinality of the target classes, and the proportion of the data that belongs to the most common class in the set.

Table 2: Datasets

Dataset	Records	Classes	Majority
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%

We generate all URL features for each of these datasets and then apply a range of machine learning techniques to build classifiers that can be analysed for key features on each task.

2.2. Feature Generation

We generate URL features using a set of functions that focus on distinct types of features applied to specific regions across the URL. This permits users to apply subsets of functions that define only the features they require. In addition we create a group of global features that contain general string properties of the entire URL, rather than specific regions. The complete set of features, along with their categorisation and definition are outlined in Table 3.

These URL feature functions are available in our open source implmentation and python package *url2features*. The package is designed such that it can process large files in chunks and will only add the specific sub-groups of features a user requests. In addition, the naming convention is designed so that features relating to specific regions of the URL can be identified and analysed as a group.

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

Listing 1: Install and invoke the url2features package from CLI

The commands for installing and invoking the *url2features* package from the command line are shown in Listing 1. When executing this code it will take the tabular data inside the file *input.csv* and then apply feature engineering on the column named *URLCOL*. The features to be created are defined by the switches $-host$ and $-tld$ for the host and top level domain features. By default the application outputs to STDOUT, so in this example we are redirecting the output of the program to create a new file *output.csv*.

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

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

Listing 2: Usage of url2features with scikit-learn pipelines

The package can also be invoked programmatically and included inside a machine learning framework like scikit-learn pipelines[30]. This is shown in Listing 2, where the same two

Table 3: URL Features

Group	Feature	Type	Definition
Protocol	protocol_name	Category	The exact internet protocol name
	protocol_type	Category	Internet protocol categorised by its purpose
	protocol_exists	Boolean	Flag indicating presence or absence of internet protocol
Host	host_is_ip	Boolean	Flag indicating if the host is an IP address
	host_has_port	Boolean	Flag indicating if a port number is explicitly specified
	domain_len	Numeric	Character length of the domain name (minus TLD and subdomain)
	domain_alpha	Numeric	Proportion of characters in the domain that are letters
	domain_sections	Numeric	Count of period separated domain sections
	subdomain	Category	The text value of the subdomain
	subdomain_type	Category	Categorisation of the purpose of the subdomain
	subdomain_freq	Numeric	Frequency of the subdomain in internet traffics
	tld_name	Category	Top Level Domain Name
	tld_type	Category	Top Level Domain extension categorised by purpose
	tld_freq	Numeric	Top Level Domain extension frequency in internet traffic
Path	path_depth	Numeric	The depth of the directory path between host and file
	path_1st_wd	Category	The first distinct word in the path longer than 2 chars
	path_1st_wd_prefix	Category	The first 3 chars of the first distinct word in the path ¿ 2 chars
	path_wd_count	Numeric	Count of distinct words in the path between host and file
	path_wd_len	Numeric	The mean length of the individual words in the path
	path_has_date	Boolean	Flag indicating if a date is detected in the path
	path_is_home	Boolean	Flag indicating if the path starts with a home directory
File	file_len	Numeric	The character length of the target file
	file_1st_wd	Category	The first distinct word in the file name
	file_1st_wd_prefix	Category	The first 3 chars of the first distinct word in the filename
	file_wd_count	Numeric	The count of words longer than 2 chars in the file name
	file_wd_len	Numeric	The mean length of the individual words in the file name
	file_ext	Category	The exact file extension e.g. ”exe” or ”php”
	file_type	Category	File extension categorised by function & purpose (e.g. Static or dynamic)
	file_ext_exists	Boolean	Flag indicating presence of a file extension for the URL target
Params	params_len	Numeric	The character length of the parameter string
	params_count	Numeric	The count of the key value pairs in the param string
	params_match	Boolean	Flag indicating if the count of keys and values in the param string match
	params_has_url	Boolean	Flag indicating presence of a raw URL within the parameter string
	params_enc_url	Boolean	Flag indicating presence of an encoded URL within the parameter string
	params_enc_char	Boolean	Flag indicating presence of an encoded character within the parameter string
	params_frag_len	Numeric	Length of a hash delineated fragement at the end of the parameter string
	keys_count	Numeric	Count of the distinct keys used in the parameter string
	keys_len	Numeric	The mean length of the individual keys in the params
	keys_numeric	Numeric	The proportion of numeric characters in the param keys
	values_count	Numeric	Count of the distinct values used in the parameter string
	values_len	Numeric	The mean length of the individual values in the params
	values_numeric	Numeric	The proportion of numeric characters in the param values
	frag_len	Numeric	The character length of the fragment string
	frag_secs	Numeric	Count of distinct sections in the fragment string delimited by [?&=]
	frag_enc_char	Boolean	Flag indicating presence of an encoded character within the fragment string

Table 4: Machine Learning Results

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

feature engineering functions (host and tld) are included as feature transformers in a machine learning pipeline. The advantage of this approach is that these transformations can be built into a single serialised model for deployment.

2.3. Feature Importance

We use the SHAP[31] package to calculate feature importance using a small holdout test set of 80-100 samples per experiment. We aggregate the Shapley values for these sample points to calculate the mean absolute contribution of each feature, as is typically done in Shapley feature importance plots.

For our URL feature analysis, we then group these individual feature importance values by the specific URL segment that the feature was derived from, using Figure 1 as a guide to these segments. We sum the feature importance for all features within a URL segment to allow us to plot the segment contributions to model performance. We provide a thin grey coloured line covering the entire segment which captures the importance of global features (like URL length). If the script is applied to a dataset with more than just URL features then all non-URL fetaures would be grouped into this global group. All segment specific URL features are plotted below their segment in a canonical example URL.

These plots allow for a structural understanding of how URL data contributes to predictive performance of a model and permits the comparison of feature importance across the six problems in this study. The script for generating these plots is provided as part of the source code for the ‘url2features‘ package.

3. Results

A set of standard machine learning pipelines are applied to each of the six problems. Each approach is applied using default parameters, without any fine tuning. We use feature preprocessing modules appropriate for each class of model. All details are available in the source repository for the experiments. The performance of the models is shown in Table 4, where we show Balanced Accuracy of each model on the holdout data, as this metric is appropriate and comparable across all six problems.

The table of results indicates that internet security problems appear much more amenable to URL based classification. All three of these problems result in high performing classifiers. By comparison the topic and user preference classification tasks achieve far more modest performance. It is worth noting that these problems are hindered by both higher cardinality in the classification targets and greater variability in dataset size, 2 of the 3 data sets are much smaller than any of the security datasets. However, the performance on the very large DMOZ dataset strongly suggests that, in general, URL based topic classification is likely to remain a difficult problem.

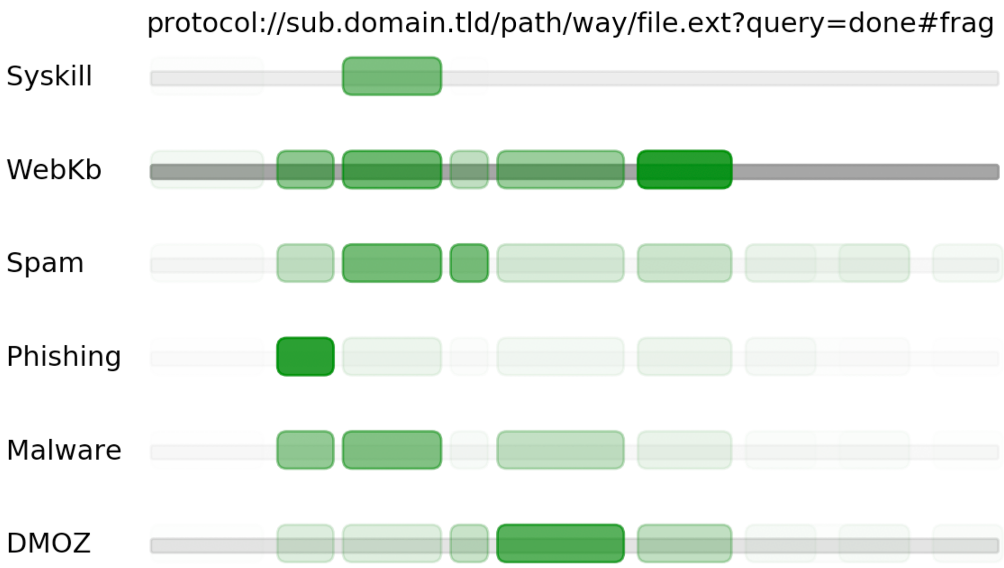

Figure 2: Structural Depiction of URL Feature Importance

We used the results from the best performing classification model for each individual problem and generated the SHAP values for the records in the test dataset. These records are then used to generate the URL Structure Feature Importance Plot shown in Figure 2. This plot aggregates the importance of individual features into their region of origin within the URL structure. Allowing us to visualise the source of signal for each task and compare features across tasks.

We see that there is substantial variability across all classification tasks in these experiments. The protocol and parameters sections remain mostly devoid of signal, with the exception of the Spam URL classification task, where we see value in the URL parameter features. The domain, path and file regions are the predominant sources of signal across these tasks. Nevertheless, the emphasis changes between them. The subdomain appears particularly important for the phishing URL identfication, whereas the URL from domain to filename is important for the WebKb classification data.

The internet security problems appear to derive less utility from the global features when compared to the three topic classification problems, as is indicated by the strength of the grey lines through the middle of the plot for each task. This suggests that these problems rely more heavily on the information from specific components of the URL structure.

The contributions of the domain remain high across all problems evaluated. However, whether the key elements is the domain name itself, the subdomain or the top level domain varies considerably across the tasks and does not appear consistant across the top broad categories of security and topic classification problems. This suggests that data scientists and machine learning engineers need to test these features on a task by task basis rather than relying on heuristics.

4. Conclusion

We have described the structural elements of a URL from which context specific features can be extracted. These features tend to be used indepedently within different problem types and tuned for specific purposes. In this paper we have described an open source

URL feature generation package that is both sensitive to the source of information within each component and sufficiently flexible for general use. We conducted experiments using the features across a range of indepenent URL classification tasks.

Our experiments demonstrated that the various structural regions play different roles across these tasks, exhibiting varying levels of importance. Some patterns (like the strength of global URL features) could be attributed to broader categories of the task, while others appear task specific. We have observed that the domain name, with its subcomponents, is a strong and consistent contributor to the performance of models across tasks. The source of information from within the domain varies, but it remains a rich source of information for machine learning tasks that rely on the URL as a central feature.

5. 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*, ICML '09, (New York, NY, USA), p. 681–688, Association for Computing Machinery, 2009.

[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.

[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), pp. 252–261, Springer Berlin Heidelberg, 06 2012.

[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), pp. 467–482, Springer International Publishing, 2016.

[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*, IWSPA '17, (New York, NY, USA), p. 55–63, Association for Computing Machinery, 2017.

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

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

[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*, ICCDA 2020, (New York, NY, USA), p. 102–106, Association for Computing Machinery, 2020.

[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*, WWW '11, (New York, NY, USA), p. 197–206, Association for Computing Machinery, 03 2011.

[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*, WWW Alt. '04, (New York, NY, USA), p. 262–263, Association for Computing Machinery, 2004.

[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*, WWW '04, (New York, NY, USA), p. 193–202, Association for Computing Machinery, 2004.

[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*, pp. 1109–1110, 01 2009.

[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, oct 2010.

[17] I. Hernández, 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)*, pp. 317–322, 2016.

[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*, CIKM '05, (New York, NY, USA), p. 325–326, Association for Computing Machinery, 01 2005.

[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, mar 2013.

[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, p. 235, jun 2014.

[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*, pp. 27–56. Cham: Springer International Publishing, 2020.

[26] G. Vrbančič, I. Fister, and V. Podgorelec, "Datasets for phishing websites detection," *Data in Brief*, vol. 33, p. 106438, 2020.

[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*, AAAI'96, p. 54–61, AAAI Press, 1996.

[29] Kaggle, "Url classification dataset [dmoz]," 2018.

[30] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, *et al.*, "Scikit-learn: Machine learning in python," *Journal of machine learning research*, vol. 12, no. Oct, pp. 2825–2830, 2011.

[31] 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.), pp. 4765–4774, Curran Associates, Inc., 2017.