J. Hlaváčová (Ed.): ITAT 2017 Proceedings, pp. 129–135
CEUR Workshop Proceedings Vol. 1885, ISSN 1613-0073, © 2017 J. Puchýř, M. Holeňa

Random-Forest-Based Analysis of URL Paths

Jakub Puchýř, Martin Holeňa

Czech Technical University, puchyjak@fit.cvut.cz
Czech Academy of Sciences, Prague, martin@cs.cas.cz

Abstract: One of the key sources of spreading malware are malicious web sites – either tricking user to install malware imitating legitimate software or, in the case of various exploit kits, initiating malware installation even without any user action. The most common technique against such web sites is blacklisting. However, it provides little to no information about new sites never seen before. Therefore, there has been important research into predicting malicious web sites based on their features. This work-in-progress paper presents a light-weight prediction method using solely lexical features of the site URL and classification by random forests. To this end, three possibilities of feature extraction have been elaborated and investigated on real-world data sets with respect to precision and recall. The obtained results indicate that there is nearly never a significant difference betrweeen the considered methods, and that in spite of the limitation to the lexical features of the site URL, they have an impressive performance in terms of area under the precision-recall curve for the path parts of URLs.
Keywords: malicious URLs detection, classification, random forest

1 Introduction

Protecting devices from malicious software is a never ending fight. There are multiple ways how malware can infect an end point device but the most common way how malware can spread itself is via malicious web sites – either tricking user to install it by imitating legitimate software or, in the case of various exploits kits, install itself without any user action or knowledge. An ideal defence against such a threat would be to block any malicious site before it can even serve its content. The most common technique, which is widely used in practice, is blacklisting. While such technique is extremely fast (just searching for the URL within the list), it provides only limited protection, since no blacklist is perfectly up-to-date [6]. In particular blacklisting provide us with little to none additional information about new, never before seen sites.

Aware of this problem security researchers have proposed various systems to protect users. We can basically divide these approaches into two groups based on information they are using. The first one is considering the content of the web site. Provos et al. [5] proposed a solution based on the position of iframes and the existence of obfuscated javascript to detect malicious landing-pages. This approach is usually more reliable because it provides all necessary informations. On the other hand, it is clearly slower since it must process more data and it is potentially dangerous since it needs to actually access the content. Zhang et al. [8] proposed method to detect phishing sites based on the TF-IDF of the whole document.

The other approach is to consider only the URL itself and information related to it, such as DNS and whois records. To this end, multiple approaches have been proposed, such as classification based on bag-of-word representation of the URL by Ma et al.[3], [4] or characters n-grams representation used by Verma et al. [7] to detect phishing sites. Both of these studies show that systems based on lexical and host based features can have very low percentage of misclassified samples; unfortunately, the studies do not differentiate between benign site classified as a malicious one and vice versa. In real-word scenarios, however, the misclassification of benign site is much more severe. Motivated by these studies, we focus on fast and light-weight classification of the URL, where we use only lexical based features of the URL and where we are trying to minimize the number of misclassified benign sites, i.e. to maximize specificity, even at the expense of a higher number of misclassified samples, ie.e, at the expense of lower accuracy. The goal of our approach is to provide classification method which would be fast and keep the number of falsely classified URLs as low as blacklisting, yet would still be able to generalize and thus be ideal as a pre-filtering method for some slower, more precise methods.

In the previously mentioned papers we can see that there are multiple different algorithms which reach very similar results, thus we decided to do the very opposite. We use only one algorithm – random forest in our case – and compare its result on different types of features generated from the same set of the URLs. To demonstrate this approach we have used a large number of URLs, labelled by a paticular antivirus software, to build our classification system. Using this data, we show that classification models based only the on the lexical parts of URL can still reach a high precision and that a different representation might be more suitable for different parts of the URL.

The following two sections recall in turn several methods for extracting features from the URL, and our algorithm of choice – random forests. After that, we describe three variants of the proposed method. These variants are experimentally compared in Section 5, after which the paper concludes.

2 Feature Extraction

Uniform Resource Locators (URLs) are the global addresses of resources on World Wide Web and they are the primary means by which users browse through the Internet. They are human-readable text strings with the structure depicted in a figure 1. The "host" is arguably the most important part of the URL. One of the main ideas, why classification based on the lexical structure of a URL might work, is that malicious URLs tend to look different. The benign URLs, especially the host parts, are mostly short and easy to remember in comparison to malicious ones.

In this section, we provide an overview of the problem and a discussion of the features used for URL classification.

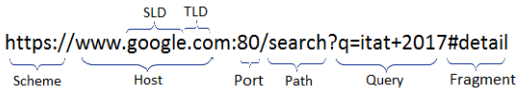

Figure 1: Example of URL and its components.

2.1 Problem Overview

A site with unsolicited content may belong to several categories such as spam, phishing, drive-by exploits, etc., but for our purpose we treat all of these categories as one, since the ultimate decision is always the same - whether to allow access to a given site, or not. We go by medical terminology which is commonly used in cyber security. Labelling a sample as a *positive* means labelling it as a malicious URL and *negative* means benign one.

We classify sites based only on the lexical structure of URLs without considering any additional information such as reputation or content. While these informations might improve accuracy, we exclude them for several reasons. First, we focus on light-weight and fast classification, thus downloading a content is out of our scope for its slowness. Second, accessing the content of a page before getting known whether it is malicious is potentially dangerous, since you might get infected before blocking the site. The reason for not using reputation or host-based information is more a technical issue since such information is fairly hard to get and the information itself might not be up to date. That is especially unsuitable in case of malicious domains, since some of them tend to exist only for few days or even hours. Nevertheless we show in Section 5 that classifying web sites using only the lexical features of the URLs can still be sufficient for a high precision.

2.2 Features Pertaining to a URL

From the point of view of classification, the most important parts are host, port, path, and query. We do not consider the fragment part since it is basically non-existent in malicious URLs and we do not use the scheme part either. One might argue, that using a secure protocol (such as https) is a strong indicator for a benign site. That is true – almost all tested malicious URL do not use secure layer. Unfortunately a lot of benign URLs, especially not very known ones, do not use it either, thus we decided to not use scheme based features to avoid skewed results.

Since our goal is to design light-weight and fast classification system, we can not afford to use complicated representation. We combine a few real-valued, hand selected features such as length of the entire URL and its parts, the number of subdomains, number of non-alphanumeric characters in a path, etc., with several text representation techniques.

1. The Bag-of-words (BoW) representation, where each part of the URL is represented as a bag of its words. While this representation is quick to compute, it suffers from two major drawbacks. The feature space is extremely large (theoretically unbounded) and the word ordering is lost which is especially troublesome in our case since we do not use any additional information but the URL. Therefore, we slightly modify the BoW to keep some additional information such as top and second level domains and last words in a path, since that is usually a file extension for a downloaded file. While models using this representation might reach high precision, they have one major drawback. Some of the malicious URLs tend to generate their domain names in a pseudo-random way, that is they generate many names which differ in a few characters, yet they still look very similar. This is the case where BoW absolutely fails as it has no way how to detect these changes.

2. To overcome this we use following technique. Character N-grams is a representation where each consecutive N characters are considered a feature. While this representation does not preserve the word ordering either, it bounds the size of feature space by $O(\Sigma^N)$ where Σ is the size of the alphabet, and to a certain extent improves BoW inability to generalize over long, pseudo-random generated words.

   While inspecting the structure of URLs, we noticed that some of the malicious URLs differ only in a few characters. Moreover, these random characters are always on the same position, thus we use a generalization of character N-grams.

3. Character N-grams with k don't care symbols is a generalization of N-grams representation, where each N-gram is transformed into a set of $\binom{N}{k}$-grams with k don't care symbols. For example, 4-grams *ITAT* and *ICAT* are transformed into following set of 4-grams with one don't care symbol: *?TAT, I?AT, IT?T, ITA?, ?CAT, IC?T, ICA?* , where "?" marks a position of the don't care symbol.

This representation is as fast to compute as the previous ones and reduces the size of the feature space even more – to $O(\binom{N}{k}\Sigma^{N-k})$. Moreover it has the best ability to generalize as we show in the results in Section 5.

3 Random Forests

We chose random forests as our classification algortihm for several reasons. It provides us with a probabilistic output, it is capable of handling a huge number of features, which is especially important in our case, and it handles naturally problems with multiple classes.

A random forest is an ensemble of randomly trained decision trees and it is characterized by multiple parameters such as the forest size T, the maximum depth D for each tree in the forest, the training objective function.

Each split node i, i.e, inner node of the tree, is associated with a binary split function

$$h(\boldsymbol{v},\boldsymbol{\Phi}_i) \in \{True, False\},$$

where $\boldsymbol{v} = (x_1, x_2, \ldots, x_d) \in \mathbb{R}^d$ denotes the feature vector and Φ_i is the optimal parameter of the i-th split node. The data arriving at the split node i are sent to its left or right child depending on the result of the value returned by the corresponding split function.

3.1 Training

The training process is typically repeated independently for each tree in a forest. During training, the optimal parameter of the split node needs to be computed for every split node in a tree. To this end, we have chosen to maximize an information gain objective function:

$$\boldsymbol{\Phi}_i^* = arg\ max_{\boldsymbol{\Phi}_i} I_i$$

where I_i is the information gain of the i-th split node, defined as:

$$I_i = H(S_i) - \sum_{j \in \{L,R\}} \frac{|S_i^j|}{|S_i|} H(S_i^j)$$

with j indexing the two child nodes. H is the (Shannon) entropy of a set S of training points:

$$H(S) = -\sum_{c \in C} p(c) \log p(c),$$

where $p(c)$ is the empirical probability based on S of the class $c \in C$.

3.2 Testing

During testing, each test point $\boldsymbol{v}$ is pushed through all trees in a forest T until it reaches a corresponding leaf. The tree predictions are combined through averaging:

$$p(c \mid v) = \frac{1}{T} \sum_{t \in T} p_t(c|v).$$

4 Proposed Approach and Its Variants

We focus on designing fast and light-weight system for URL classification, which uses only a minimal amount of available information about the given site. With such a small amount of information, it is hardly possible to reach 100% accuracy of classification. Aware of this problem, we suggest our classification system to be used as a pre-filter for some method using a broader spectrum of features. Hence, our goal is to design a system with extremely low number of false positives, thus reaching high precision. Since we design our system as a pre-filter, any false-positive result is a final one whereas false-negative results might be adjusted by some of the later methods. Therefore, we decided to prioritize precision over accuracy.

We describe two slightly different systems which differ only in the last step - the way how the final classification is produced. The feature extraction is common to both systems and it is executed in the way described in Section 2.2.

The first, common approach for building such a system is to simply train a classifier (random forest in our case) on the gathered data and set the threshold high enough to minimize the number of false positives. We propose another solution that is based on the fact that each part of the URL (i.e. host, path, or query) has a different lexical structure and wording. Therefore, we train specialized models for each part of the URL separately and the final model is a team of these. The final classification is done via a voting of all models, where the URL is classified as malicious if at least one of the models classified it so. An advantage of using separate models for each part is that it reduces the dimensionality and overall size of the model. That is especially case for the host part, since there are usually multiple URLs that differ only in the path or query part. Thanks to this, we can afford to either train more complex models or to retrain them more frequently, and thus indirectly increase accuracy of our models.

5 Experimental Evaluation

In this section we describe experiments that were performed to test and compare the proposed methods. We evaluated the methods with two performance measures. Both of them measure the performance of a whole family of classifiers corresponding to some finite set Θ of decision thresholds. The first one is an area under the precision-recall curve, where precision and recall corresponding to a threshold Θ are defined the standard way, i.e.

$$precision = \frac{tp(\Theta)}{tp(\Theta) + fp(\Theta)}$$
$$recall = \frac{tp(\Theta)}{tp(\Theta) + fn(\Theta)}$$

where $tp(\theta), fp(\theta), tn(\theta)$, and $fn(\theta)$ denote the number of true positives, false positives, true negatives and

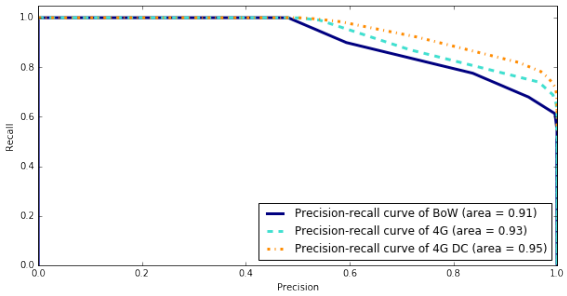

Figure 2: Precision-recall curve for host based features.

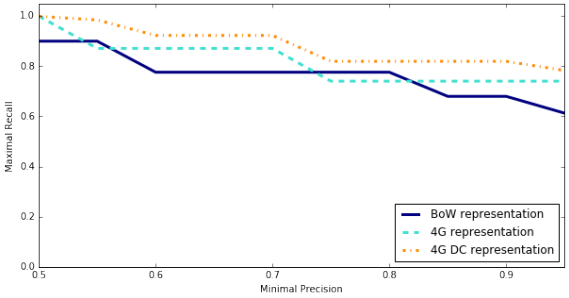

Figure 3: Rho function for host based features.

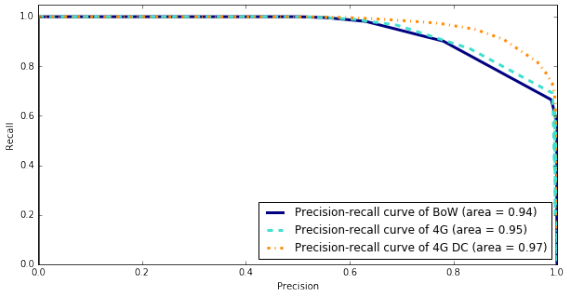

Figure 4: f-recall curve for path based features.

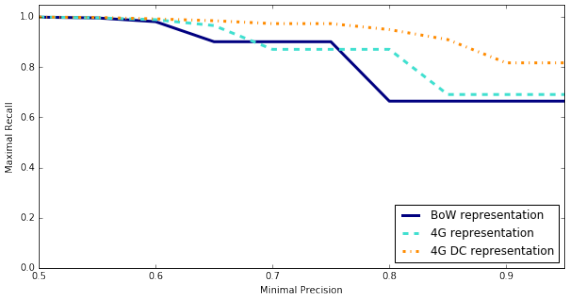

Figure 5: Rho function for path based features.

Table 1: Size of the small datasets and size of feature space

TLD	URLs	BoW	4G	4GDC	TLD	URLs	BoW	4G	4GDC
com	180381	323785	164395	197993	eu	1845	10863	2058	24086
ru	38281	115616	37059	112531	ua	1732	9970	1832	22460
net	26140	91532	26319	109033	in	1676	9872	1791	23592
com.br	10687	31055	10595	40173	ro	1480	7811	1592	17095
org	10469	51784	11139	80083	biz	1361	8801	1598	21942
info	6036	33933	6788	60505	cc	1151	7063	1255	19328
de	5791	28068	6148	44288	us	1090	6462	1212	16487
it	4939	20732	5081	34099	me	1034	5935	1144	16098
lt	3688	12085	3746	25337	com.au	955	6383	1066	14857
co.uk	2290	12804	2437	24812					

false negatives for the classifier corresponding to the threshold θ.

The second one the maximal recall among classifiers from the considered family that have precision above fixed limit:

$$\rho(l) = \max\{\frac{tp(\theta)}{tp(\theta)+fn(\theta)} | \theta \in \Theta, \frac{tp(\theta)}{tp(\theta)+fp(\theta)} \geq l\}$$

We evaluated each of the proposed feature generation techniques in the following way. For each method we used a separate balanced dataset which contained the same set of labelled URLs and then trained the several random forests with different number of trees T and maximum depth D. We always used 80% of data for training and 20% for testing. For the N-grams representations we chose to use 4-grams and 4-grams with 1 don't care symbol which we shorten as 4G and 4GDC respectively in the following sections.

Table 2: Number of URLs in each dataset and dimensionality of the corresponding feature space for each of the considered feature extraction methods.

Data	URLs	BoW	4G	4GDC
Host	508074	452 950	977 754	223 281
Path	2732415	3 559 094	5 264 007	1 875 289

5.1 Employed Data

We used the set of labelled URLs provided by one of the antivirus companies, which were collected over the period of one month.

We didn't explicitly extract URLs related to one type of threat (such as URLs related to malicious/phishing campaigns) and thus the URLs structures are partially unrelated. It is because URLs of different malicious campaigns differs from each other even though they are labelled the same – as a malicious. For example URLs related to a phishing campaign targeting facebook will have similar structure but they will be different from URLs connected to C&C servers of some particular botnet.

To test all suggested methods, we generated multiple datasets from it. Namely we created several small datasets from host parts of the URLs to test statistical significance of our methods, where each dataset contains only URLs with the same top-level domains (such as *'.com'*) or country-code second-level domains (such as *'co.uk'*).

The number of URLs in each dataset and the dimensionality of the corresponding feature space for each of the considered feature extraction methods are in table 1.

Next, we used datasets for host and path URL-parts and for each considered method of feature extraction. Table 2 contains the number of URLs in each dataset and the dimensionality of the corresponding feature space for each of the considered feature extraction methods.

5.2 Main Results

The comparison of the three considered feature extraction methods for the area under the precision-recall curve and for the maximal recall of classifiers with precision above 0.8 is for 4 different combinations of the forest size and tree depth summarized in Tables 6 and 7, respectively. The considered family of classifiers was constructed using the 20 thresholds 0.05, 0.1,...,1. For each of those 4 combinations, the hypotheses that all three methods lead to the same area under the precision-recall curve and to the same maximal recall of classifiers with precision above 0.8 were tested with the Friedman test [1]. The results of their testing are given in Tables 3 and 4, respectively.

Table 3: Friedman test for a maximal recall function for different values of T and D. The exact values are shown in each column for T and D respectively.

values of T D	10 200	10 50	50 200	50 50
χ^2 statistic	2.9500	2	0.1100	8.3200
p-value	0.2300	0.3700	0.9500	0.0160

Table 4: Friedman test for an area under precision recall curve for different values of T and D. The exact values are shown in each column for T and D respectively.

values of T D	10 200	10 50	50 200	50 50
χ^2 statistic	1.2600	2.2100	1.3700	0.3200
p-value	0.5300	0.3300	0.5000	0.8500

The test rejected only the hypothesis of the same maximal recall for a single considered combination tree depth 50 + forest size 50. For that combination, we performed the posthoc test of equal mean ranks with respect to the maximal recall for the 3 possible pairs of the compared methods [1]. The results of that posthoc test are presented in Table 5.

Table 5: Post hoc test for maximum recall function for random forest with $T = 50$ and $D = 50$.

	4G-4GDC	4G-BoW	4GDC-BoW
Statistic	2.1089	2.7578	0.6489
p-value	0.0175	0.0029	0.2582

They imply that the hypothesis of equal mean ranks with respect to the maximal recall is on the significance level 0.05 rejected for the pairs of methods 4G–4GDC and 4G–BoW, but not for the pair 4GDC–BoW. This implication is valid for several kinds of corrections with respect to the simultaneous testing of all three pairwise hypotheses, e.g., the correction according to Holm, the correction according to Shaffer, and the most general correction according to Bergmann and Hommel [2].

In addition, we compared the considered feature extraction methods on the host and path part of URLs. For comparability of results, we chose the same random forest parameters as before. Namely we used the number of trees $T = 20$ and maximum depth $D = 100$.

For classification using only the host part of the URLs, the results show, that the techniques rank from the best as follows: 4-grams with one don't care symbol, 4-grams, BoW. All methods are able to reach a high precision, but it can be seen that BoW generalize poorly on features based on the host part only. It is not surprising, since all of the URL in the dataset were unique from each other, thus BoW could reliably detect only URLs for which at least some subdomains of the URL were shared by the training and testing part of the dataset.

When we used only path parts of URLs, the results were similar. Surprisingly, the BoW representation performed almost as well as 4-grams while 4-grams with don't care symbol significantly improved recall for high thresholds. Although we expected this results with representation using don't care symbols, we can not explain why 4-grams have not outperformed BoW.

6 Conclusion

This work-in-progress paper is a small contribution to research into predicting malicious web sites. It presented a light-weight approach using solely lexical features of the site URL and classification by random forests. Three methods for the extraction of such URL features have been considered and experimentally validated on real-world data. The data included on the one hand smaller datasets containing the host parts of URLs from 19 mutually unrelated (mostly top-level) domains, on the other

Table 6: Comparison of the area under the precision-recall curve between the considered feature extraction methods for particular combinations of random forest parameters T and D. The comparison is based on the datasets from the 19 domains listed in Table 1, each value shows how frequently the method in the row yielded a higher area under the precision-recall curve than the column method.

>	4G	4GDC	BoW
4G	-	7	8
4GDC	12	-	9
BoW	11	10	-

(a) $T = 10$ and $D = 200$

>	4G	4GDC	BoW
4G	-	7	7
4GDC	12	-	8
BoW	11	11	-

(b) $T = 10$ and $D = 50$

>	4G	4GDC	BoW
4G	-	10	8
4GDC	9	-	7
BoW	11	12	-

(c) $T = 50$ and $D = 200$

>	4G	4GDC	BoW
4G	-	9	11
4GDC	10	-	10
BoW	8	9	-

(d) $T = 50$ and $D = 50$

hand one large dataset containing the host parts and one large dataset containing the path parts of URLs from a mixture of many top-level domains. All dataset were specific for each considered feature extraction method. The considered methods were compared with respect to the area under the precision-recall curve and with respect to the maximal recall of classifiers with precision above 0.8. The smaller datasets with the host parts of URLs alowed for testing differences between the methods. No significant differences between the methods have been found with respect to the area under the precision-recall curve, whereas with respect to the maximal recall of classifiers with given precision, significant differences were only for one combination of the forest size and tree depth, and only between 4-grams and 4-grams with one don't care symbol, and 4-grams and the bag-of-words representation. Finally, the results obtained on the large datasets show that in spite of the restriction to lexical features, all three methods achieve quite impressive area under the precision-recall curve for the path parts of URLs.

Acknowledgement

The research reported in this paper has been supported by the Czech Science Foundation (GAČR) grant 17-01251.

References

[1] J. Demšar. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research*, 7:1–30, 2006.

[2] S. Garcia and F. Herrera. An extension on Statistical Comparisons of Classifiers over Multiple Data Sets for all pairwise comparisons. *Journal of Machine Learning Research*, 9:2677–2694, 2008.

[3] Justin Ma, Lawrence K. Saul, Stefan Savage, and Geoffrey M. Voelker. Beyond blacklists: Learning to detect malicious web sites from suspicious urls. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '09, pages 1245–1254, New York, NY, USA, 2009. ACM.

[4] Justin Ma, Lawrence K. Saul, Stefan Savage, and Geoffrey 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, pages 681–688, New York, NY, USA, 2009. ACM.

Table 7: Comparison of the maximal recall of classifiers with precision above 0.8 between the considered feature extraction methods for particular combinations of T and D. The comparison is based on datasets from the 19 domains listed in Table 1, each value shows how frequently the method in the row yielded a higher maximal recall of classifiers with precision above 0.8 than the column method.

>	4G	4GDC	BoW
4G	-	9	6
4GDC	10	-	7
BoW	13	12	-

(a) $T = 10$ and $D = 200$

>	4G	4GDC	BoW
4G	-	11	6
4GDC	8	-	6
BoW	13	11	-

(b) $T = 10$ and $D = 50$

>	4G	4GDC	BoW
4G	-	9	9
4GDC	10	-	9
BoW	10	10	-

(c) $T = 50$ and $D = 200$

>	4G	4GDC	BoW
4G	-	5	4
4GDC	14	-	8
BoW	15	11	-

(d) $T = 50$ and $D = 50$

[5] Niels Provos, Panayiotis Mavrommatis, Moheeb Abu Rajab, and Fabian Monrose. All your iframes point to us. In *Proceedings of the 17th Conference on Security Symposium*, SS'08, pages 1–15, Berkeley, CA, USA, 2008. USENIX Association.

[6] Sushant Sinha, Michael Bailey, and Farnam Jahanian. *Shades of Grey: On the effectiveness of reputation-based blacklists*, pages 57–64. 2008.

[7] Rakesh Verma and Avisha 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, pages 55–63, New York, NY, USA, 2017. ACM.

[8] Yue Zhang, Jason I. Hong, and Lorrie F. Cranor. Cantina: A content-based approach to detecting phishing web sites. In *Proceedings of the 16th International Conference on World Wide Web*, WWW '07, pages 639–648, New York, NY, USA, 2007. ACM.