Request Type Prediction for Web Robot and Internet of Things Traffic

H. Nathan Rude and Derek Doran
Department of Computer Science & Engineering
Kno.e.sis Research Center
Wright State University, Dayton, OH, 45435
{howard.rude, derek.doran}@wright.edu

***Abstract*—The volume of Web robot traffic seen by Web servers and clouds continue to increase with the popularity of Internet of Things (IoT) devices. Such traffic exhibits decidedly different statistical and resource request patterns compared to humans. However, the optimizations ensuring high levels of Web systems and cloud performance requires traffic to exhibit the statistical and behavioral patterns of humans, not robots. This necessitates the design of novel Web system optimizations to handle Web robot traffic effectively. Caches are a basic component of high performing Web systems, but their effectiveness relies on accurate resource request prediction. In this paper, we explore a suite of classifiers for the resource request type prediction problem for robot traffic. Our analysis reveals: (i) a striking difference in the request patterns of robots across multiple servers from the same domain; and (ii) that Elman neural networks hold promise to predict request types despite these differences.**

I. Introduction and Motivation

The proportion of traffic that can be attributed to Web robots, defined as *any* automated agent (including software agents and hardware IoT devices) that submits http requests to a Web server automatically and without any human intervention, has risen to staggering levels. Whereas decade old studies found approximately 20% of requests to a Web server are from robots, recent measurements identify over half of all requests submitted to academic, e-commerce, staging, and vertical search engine sites, and over 60% of traffic across a swath of public Web systems to be generated by these automatic crawlers [1]. This sharp rise is precipitated by the proliferation of Web 3.0 technologies that encourage users to share valuable in-the-moment thoughts and social data [2]. IoT devices that send automated requests to Web servers and services that stand to further raise this proportion in the future.

The most basic driver for the performance of a Web server may be a Web cache, which provide low response rates to client requests [3], reduce the number of bottlenecks on a network [4], and are instrumental for building scalable server clusters [5]. However, prevalent Web caching architectures and polices were developed without considering the possible impact of Web robot traffic, which features differentiated functionality [6], access patterns [7], and traffic characteristics [8]. Previous studies have noted that a cache's hit ratio is reduced by Web robot traffic [9], supporting the notion that the intrinsic differences in their traffic compared to humans [6], [7] make their requests incompatible with current Web caches. The effect of an incompatible cache may be very significant: if increases in Web robot traffic cause linear decreases in a cache's hit ratio from A to B, its performance is identical to a cache facing no robot traffic and with a hit ratio of A, but has an exponentially smaller size [10].

Given that robot traffic levels on the Internet have reached never before seen heights and exhibit request patterns that are decidedly different compared to humans [7], contemporary Web caches that predict the next request made by a session [11] are unlikely to be effective. Tantamount to the construction of caches that yield high hit ratios, therefore, is a method to predict the next resource requested by a Web robot or IoT device. This paper explores the performance of numerous machine learning classifiers for this purpose. Data from multiple time periods and academic servers show how, in contrast to what may be a prevailing assumption, the sequence of resource type access patterns by Web robots are different among Web servers even if they serve the same domain of the Web. Moreover, we find that the features of a resource request sequence matter more than the order of the requests when making an accurate request type prediction, and that the performance of most classifiers are very sensitive to the nature of robot traffic to a Web server, even when the servers are from the same domain. But Elman neural networks, which consider both the frequency of recent resource types requested *and* the order of such requests yield strong, consistent performance. The findings ultimately suggest that Web caches are capable of anticipating the types of resources Web robots will request with low computational overhead, paving the way for the development of new caching algorithms able to serve Web robot and human traffic.

The layout of this paper is as follows: Section II discusses the notion of a resource request type. Section III discusses the classification algorithms used for our analysis. Section IV describes the datasets used in our experiments. Section V discusses the results of the comparative analysis. Related research is presented in Section VI. Conclusions and future work is offered in Section VII.

II. Resource Request Types

Ideally, Web caches should be equipped to predict the precise request that will be requested next by an active Web robot session. However, this prediction task may be infeasible given

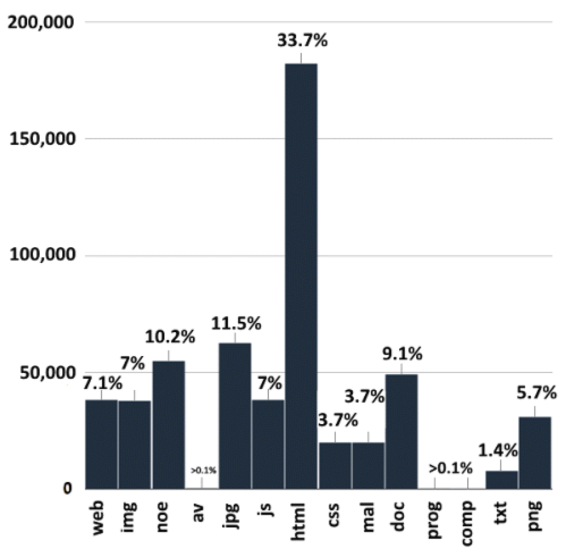
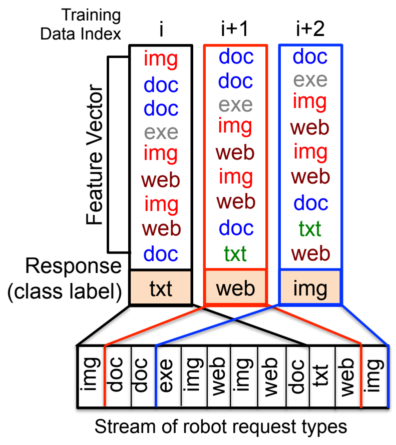


Fig. 1: Request type distribution to the WSU Web server

Class	Extensions
text	txt, xml, sty, tex, cpp, java
web	asp, jsp, cgi, php
html	html, htm
css	css
js	js
img	tiff, ico, raw, pgm, gif, bmp
png	png
jpg	jpeg, jpg
doc	xls, xlsx, doc, docx, ppt, pptx, pdf, ps, dvi
av	avi, mp3, wvm, mpg, wmv, wav
prog	exe, dll, dat, msi, jar
compressed	zip, rar, gzip, tar, gz, 7z
malformed	request strings that are not well-formed
noExtension	request for directory contents

TABLE I: Representative resources in each class



Training Data Index
i
i+1
i+2
Feature Vector
Response (class label)
img doc doc exe img web img web doc
txt
doc doc exe img web img web doc txt
web
doc exe img web img web doc txt web
img
img doc doc exe img web img web doc txt web img
Stream of robot request types

Fig. 2: Extracting samples from a resource request pattern

the large set of possible resources present on a Web server and because robots have the potential to request such resources in a seemingly random order. Even the task of predicting the specific type of resource extension may require a model to predict one outcome out of hundreds, which is challenging for lightweight classifiers that need to make decisions in the moments between http resource requests in a session. Inspired by our past work that was able to accurately separate Web robots from humans in Web logs based on meaningful pattens in the stream *resource type* requests from robots [12], this work considers classifiers that predict resource types. This prediction may be very useful for predictive web caching. For example. because the popularity of robot requests exhibit a power tail [8], the most popular resources of a predicted type are the ones likely to be requested next.

Following our previous work, we partitioned the set of resources served by a web server into the classes featured in Table I. We further expanded these partitions by separating from the web class the specific resources css, and js and the img class the resources $jpeg$ and png. We rationalize this additional division by examining the distribution request types on the Wright State University (WSU) Web servers from May 1st to July 31st 2015 in Figure 1. It shows that the $html$, css, js, $jpeg$, and png file types each represent a significant percentage of the resources requested on a Web server. Of all img type files, png and jpg requests represent 23.5% and 49.5% of the population respectively. Similarly for web files, $html$, js and css represent 65.4%, 13.6% and 7.1% of the population respectively. Since these specific resources represent a large number of requests, separating them out from the more general web and img classes enables the predictor to predict more specific resource types. Such specificity enables a Web cache that admits resources based on request type predictions to choose from among a smaller set of resources.

III. Classification Algorithms

To predict the type of an incoming request sent by a Web robot, we consider classification algorithms that try to predict the type of the n^{th} resource requested during a Web robot session given a sequence of the past $n-1$ request types. A record of the training set is denoted $\mathbf{r}_i = (v_i, l_i)$ where v_i is an ordered sequence of the past $n-1$ request types and $l_i = x_n$ is the type of the resource requested after the sequence v_i. Derivation of such sequences from a Web server access log is illustrated in Figure 2 with $n = 10$. The first record is composed of the first nine requests and its class label is given by the tenth request; the second record contains the second through tenth request and its class label is given by the eleventh request; and the third record contains the third through eleventh request and its class label is the type of the twelfth request. The trained predictor will maintain a history of the types of the last $n-1$ requests of a Web robot session and, based on this history, generate a probability distribution for the type of request that will be made next.

The selection of resource request type sequences as our training data is intentional. Although other features such as the standard deviation of requested page depth, the percentage of consecutive sequential http requests, the html-to-image ratio, and the percentage of 4xx error responses are robot session features used in previous machine learning classifiers [13], they may not be indicative of the future resource request behavior of Web robots. This is because the kinds of resources requested by a Web robot are likely to be independent of session level features. For example, whether or not a robot has a penchant for html or image files, or if a robot generates many 4xx error responses, may have little to do with its decision for the next type of resource it requests. It is instead the decisions about what resources should be requested and the way it chooses to crawl a Web site's structure (specified a priori by its programming) that is central to a resource request prediction. Such decisions are encoded in the sequence of past resource requests, not session summary statistics.

We next present the set of classifiers that were chosen for request type prediction. A special focus is given to the Elman neural network (ENN) classifier, a recurrent neural network that we hypothesize will work best for request type prediction.

A. Elman neural networks

We hypothesize that both the *features* of a request sequence as well as the *order* of types carry important information for predicting the type of the subsequent request. For example, consider the two request sequences $\langle web, web, img, img, web\rangle$ and $\langle web, img, web, img, web\rangle$. On the one hand, both sequences exhibit similar features, namely, that they are composed of 3 web and 2 img requests. These features may imply that the next request will be either for a web or img resource. But recognizing that requests for the same resource come in pairs in the first sequence and that the request types are alternating in the second, we may predict that the type of the next request will be web in the first and img in the second pattern respectively. A promising predictive model for our problem should thus be able to find associations between both the *features* of a request (e.g. the number and kinds of request types present), as well as the *order* in which these requests are in. An ENN is a classification model able to satisfy these requirements [14]. Illustrated in Figure 3, an ENN is a feed-forward neural network with a single hidden (blue) and recurrent (red) layer. Each hidden unit corresponds to a logistic function whose parameters are fitted so that the function can identify the presence of an implicit pattern or characteristic in the input data. For example, the training phase may fit the parameters of a hidden unit layer to 'activate', that is, return a value close to 1, when a request sequence contains only web or img requests. Another separate hidden unit may activate if doc requests are dominant in the sequence.

Importantly, an ENN is augmented with a recurrent *context* layer (the red units in Figure 3) that, during the training phase, store the output of hidden layer nodes activated during the training process. These values are fed back into the same hidden layer node during training of the subsequent request. This recurrence effectively tunes the parameters of the hidden unit to not only the features of a training record, but also to the ordering of requests in previously seen records. To conceptualize this, consider the diagram of an ENN with three hidden states in Figure 3. During training on record $\mathbf{r}_i$, the

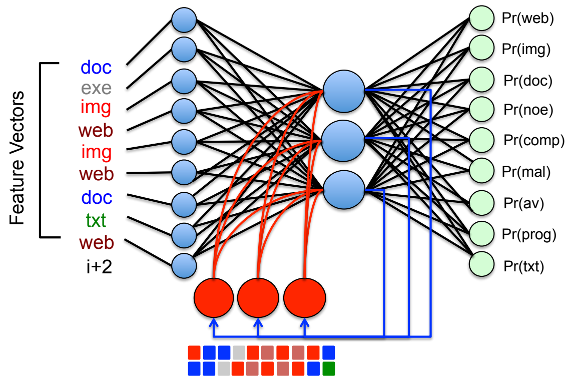


Fig. 3: Elmen Neural Network

hidden units use v_i to emit a value based on their present parameters, which are routed to the output units as well as to a context unit. The hidden unit parameters are then adjusted according to the error between the networks prediction and the actual class label l_i. When the network begins its learning process for training sample $\mathbf{r}_{i+1}$, the hidden nodes now emit a value based on v_{i+1} as well as its previous activation value from v_i through the recurrent context layer. Since this previous activation value represents a hidden unit's computation from when an img type in the first position of feature vector v_i, a doc in the second position of v_i, and so forth, the hidden node's new value will consider the state of previous request sequences as well. Similarly, when the ENN begins training over record $\mathbf{r}_{i+2}$, the context layer incorporates the state of the hidden units from the previous two training vectors.

B. Comparative algorithms

The relative importance of request pattern features and its order is unknown a priori. In order to evaluate our hypothesis that both the *features* and the *order* of a resource request pattern is important, we consider alternate classifiers to compare the performance of the ENN against that focus on either aspects, but not both. These classifiers, implemented through the WEKA Java library [15] are:

- **Discrete time Markov chain (DTMC)** [16]: *order-based*. A DTMC encodes the probability that a resource request of type j is requested following a request for a resource of type i. The model is 'memoryless', in the sense that the predictions are only based on the type of the last request seen. It thus captures the order of a request sequence without regard to features such as the frequency different types of resources emerge.
- **Multinomial logistic regression (MLR)** [17]: *feature-based*. The MLR trains an ensemble of binary logistic regression models, one for each pair of possible outcomes. One model is chosen as a "pivot" and every other model is compared to this pivot. The predicted class is the outcome selected by highest accuracy.
- **J48 decision tree (J48)** [18]: *feature-based*. The J48 is an implementation of the C4.5 algorithm to build a decision tree. Each node of the tree uses the attribute

with the maximum information gain that best separates each subset of the tree into classes.

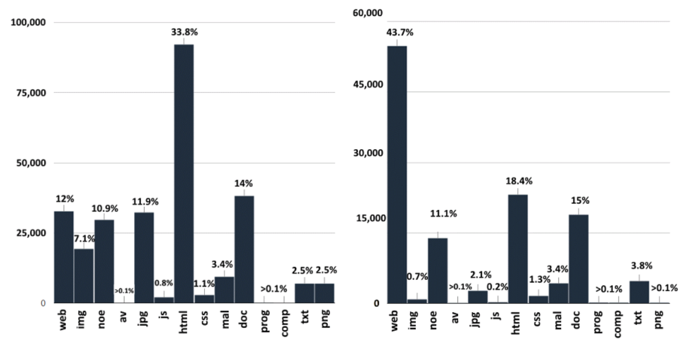
- **Support vector machine (SVM)** [19]: *feature-based*. A SVM maps the data into a k-dimensional space and uses a hyperplane to split the data points while maximizing the margin between points of different classes. We use the default radial basis function (RBF) kernel for our experiments. The RBF kernel enables fast evaluation and provides fewer terms for approximation [20].
- **Näive Bayes (NB)** [21]: *feature-based*. NB uses prior observations to calculate the probability that a feature will occur given a class. The classification of an observation is given by the product of the probabilities of each feature occurring, given a class label.
- **Feed forward neural network (NN)** [22]: *feature-based*. The NN is the basis for the ENN shown in Figure 3, but because it lacks a context layer, does not consider the order of resource type sequences.

IV. Dataset Description

This section describes the datasets used to compare the performance of the classifiers for request type prediction. They capture Web robot requests from Web servers across two different universities, namely Wright State University (WSU) and the University of Connecticut (UConn) over different time periods. Despite the fact that both datasets come from an academic domain, we demonstrate their exposure to Web robot traffic with differing request type patterns. We describe the identification of Web robot sessions and contrast the resource request activity of Web robots in them.

A. Data Collection

The WSU dataset contains a record of over 244,000 requests sent to its public facing Web servers between May 1st and July 31st 2015. Because these requests include both human and robot requests, a filtering step is necessary to extract only Web robot requests. We note that numerous Web robot detection methods exist [23], however, some admit a small amount of error. Because we need a dataset consisting of *verifiable* Web robot requests to compare classification algorithms, we adopt a heuristic procedure that labels a session as a robot from a curated knowledge base of Web robot user-agent fields. Specifically, we submit queries to the website *www.botsvsbrowsers.com*, an online database containing data from over 1.4 million reported web user-agents and IP addresses from Web server administrators across the world. Since there is no public API available, we submit http requests to url pages using Python's BeautifulSoup [24] http parsing library. Each http request will return a list of user-agents, tagged as bot or $browser$, that approximately match the query. Because the site returns a set of user-agent fields that are similar to the query, there is the possibility that both human and robot user-agents are present. We thus conservatively label a user-agent as a robot only if more than 75% of the user-agents returned by the database is tagged as bot. The identified Web robot requests are then ordered by time, IP address, and user-agent to group them into sessions with a 30 minute timeout [9]. This leads to a total of 54,371 robot sessions.



(a) WSU (b) UConn

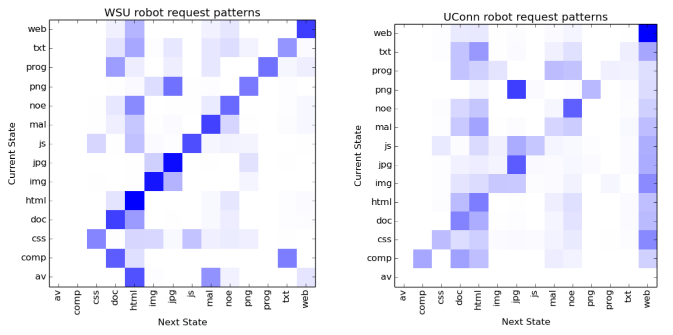
Fig. 4: Web robot resource type request distribution

The second dataset consisted of Web robot requests captured in the Web logs of the School of Engineering at UConn. This data was provided as a text file of all Web robot requests from the data providers. The robot requests in this dataset was derived from a preprocessing routine involving a blacklist of well known Web robot user-agent strings [12]. Although the two datasets emerged from different processing routines, both approaches filter away any requests that are not verifiably generated by a Web robot. The UConn data set features over 125,000 robot requests captured between January 30th and February 12th of 2009.

B. Robot activity comparison

We now summarize and compare the request type patterns exhibited by Web robots in the two datasets. Figure 4 compares the distribution of the resource types requested by robots. It shows how the most popular resource types in the WSU dataset are $html$ (33.8%), followed by doc (14%), web (12%), and jpg (11.9%). Resource type favoritism in the UConn data is very different, however, with web as the most popular resource type (43.7%), followed by $html$ (18.4%) and doc (15%). These differences may arise because, although both datasets come from an academic domain, resource request types are driven by the functionality of a Web robot [6]. This suggests that robots with varying functionality and interests may be crawling these two academic Web sites.

We also summarize the request type sequences exhibited by robots in the two servers using Figure 5. It shows the transition probabilities of a DTMC model fitted to all Web robot requests seen in the WSU and UConn logs, respectively, with bolder colors corresponding to higher probability transitions. For the WSU Web servers (Figure 5a), we find that the probability of consecutive requests for the same type of resource is significant for every type of resource. Moreover, $html$ files are seen to have a non-negligible probability of being requested following any resource type request. This is not surprising since $html$ files represent a large proportion of the dataset. In contrast, the types of resources requested by Web robots in the



(a) WSU (b) UConn

Fig. 5: Request type patterns of Web robots

UConn data (Figure 5b) show a lower probability of requesting the same resource type consecutively. It is important to note how Web robots to exhibit different request patterns on the two different academic Web servers. This note lets us evaluate the types of classifiers that will perform best across datasets that feature varying resource request patterns.

V. Results and Discussion

This section compares the performance of the collection of classifiers for prediction resource request types over the two Web server logs. For both logs, only sessions with length $k \geq 10$ are considered. We chose k based on a robot detection study which found that $k \geq 10$ was sufficient to identify the session as human or robot using resource request patterns [12]. This resulted in collecting 117,114 request sequences from the WSU logs and 93,539 request sequences from the UConn logs. We down sampled to 90,000 sequences so that an identical number are used to evaluate the Web servers. Training was performed over the oldest 70% of the sequences and testing with 30% of the most recent sequences.

Figure 6 shows the accuracy of the seven classifiers. Their prediction accuracy is similarly high across the WSU data, but shows more variability with the UConn data. Interestingly, the DTMC *order-based* classifier performs the worst over the WSU data set. This is surprising because the first-order resource request pattern visualized in Figure 5a appear highly predictable: the next resource type is often identical to the type of the last request. Moreover, the ENN, which is *order-* and *feature-based*, performs nearly as well as purely *feature-based* classifiers. This suggests that request sequence features are more important than the order in which resources are requested. However, in the UConn results, *order-based* algorithms performed very well with the DTMC yielding accuracy comparable to the *feature-based* classifiers. The ENN gives very strong results, performing 13.4% higher than the best *feature-based* classifier (SVM).

Our analysis suggests that, no matter the dataset, the $features$ of a resource request sequence matter the most to make an accurate request type prediction. Moreover, the *order* of these resource type requests may significantly improve the accuracy of *feature-based* prediction as demonstrated by the

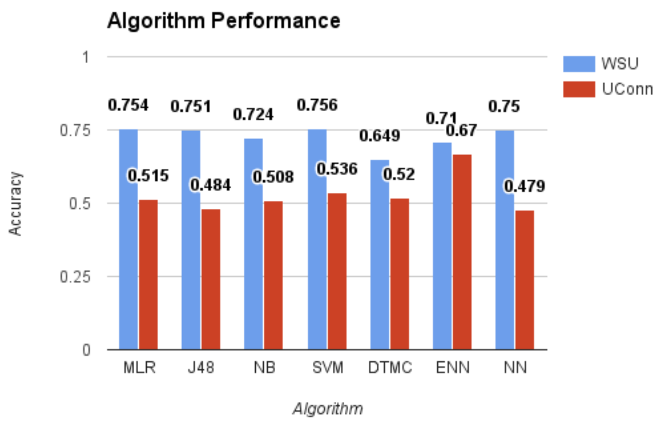


Fig. 6: Prediction Accuracy

ENN's performance over the UConn dataset, or marginally reduce performance as seen by the WSU dataset. We hypothesize that there exists a connection between the complexity of the request type sequences and the effectiveness of an *order-based* classifier. For example, the request type patterns of robots over the WSU dataset (Figure 5a) exhibit a very simple structure. The simplicity of this structure suggests that there is little information that can be gleamed from studying the order of resource requests. For example, the DTMC principally learns that the same type of resource is often requested consecutively over the WSU data set, but such a rule is a coarse summarization of how robots decide to submit resource requests. The higher complexity of resource request order in the UConn dataset (Figure 5b), however, encodes a larger amount of information that an *order-based* classifier can use to learn more specific, telling patterns in Web robot request sequences. Fortunately, the ENN model appears to discount the order of request sequences when they are insignificant (leading to only marginally worse performance over the WSU data) or promote them if they carry predictive power (leading to significantly better performance over the UConn data). This suggests that the ENN thus shows promise as a generalizable predictor of request type patterns for Web robot traffic.

VI. Related Work

Previous studies of Web robot traffic concentrate on the characterization of their traffic and on the application of machine learning methods to predict Web robot activity. Characterizations of Web robot traffic emphasize summarizing and fitting the request- and session-level features of their traffic to statistical distributions. For example, Dikaiakos *et al.* revealed that heavy-tailed distributions exist in the inter-arrival times of Web robot requests, and that robots generate 4xx response codes at a rate nearly twice that of the total Web client population [25]. Our past work [6], [7] also confirm the presence of heavy-tailed patterns in the response and inter-session times of Web robot requests. However, less work has been performed to understand the behavioral patterns of Web robots, particularly with respect to patterns in their resource type requests. In this work, we present a look into these seldom studied request patterns across multiple Web servers.

Machine learning methods have been applied to Web robot traffic for detection purposes. Stevanovic *et al.* argue that two features, namely the standard deviation of requested page-depth and the percentage of consecutive sequential HTTP requests belonging to the same web directory, are essential to separate robots from humans in Web server logs [13]. Yang *et al.* also consider association rule mining to create an n-gram model of occurrence frequencies. The n-gram model encodes adjacent sequences (substrings) of items accessed just before a prediction, prunes substrings with a low support, and uses the remainder to build a prediction model [11]. With a large amount of work already devoted to the Web robot detection problem [23], this work takes the next step: given a way to separate out human and robot requests from a Web server log, it uses applied machine learning methods to improve the capability of Web systems to handle the detected sessions.

VII. Conclusions and Future Work

This paper presented a comparative analysis of classification algorithms for predicting the next type of resource requested by a web robot. Our findings conclude that $features$ of a resource request sequence are more important than the *order* of the request sequence when making an accurate prediction for the next resource type. Furthermore incorporating *order* of a request sequence to a *feature-based* prediction may significantly improve the prediction accuracy (seen in UConn data) or marginally reduce the accuracy (seen in WSU data). The performance of *order-based* classifiers may be correlated to the complexity of the resource request patterns of a dataset. Moreover, we conclude that the ENN, an algorithm that considers the *order* and $features$ of request sequences, yields accurate and consistent performance irrespective of the nature of the robot activity on the web server.

Future work will further explore how *feature-* and *order*-based features can be incorporated to predict resource types, and whether deeper neural networks with larger recurrent layers can improve classification performance. We will also use these results to build a new predictive caching system for Web robot traffic, and devise systems to integrate both human and robot caches together to yield superior caching performance for Web servers, data centers, and clouds.

Acknowledgement

We are grateful for the support from Logan Rickert, whose data collection and archiving, Web log preprocessing, and system administration tasks supported this work. This paper is based on work supported by the National Science Foundation (NSF) under Grant No. 1464104. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the NSF.

References

[1] "Report: Bot traffic is up to 61.5% of all website traffic," bit.ly/MoMRxE.

[2] E. Qualman, *Socialnomics: How social media transforms the way we live and do business*. John Wiley & Sons, 2012.

[3] J. Wang, "A survey of web caching schemes for the internet," *ACM SIGCOMM Computer Communication Review*, vol. 29, no. 5, pp. 36–46, 1999.

[4] C. Aggarwal, J. L. Wolf, and P. S. Yu, "Caching on the world wide web," *IEEE Transactions on Knowledge and Data Engineering*, vol. 11, no. 1, pp. 94–107, 1999.

[5] K. Rajamani and C. Lefurgy, "On evaluating request-distribution schemes for saving energy in server clusters," in *IEEE International Symposium on Performance Analysis of Systems and Software*, 2003, pp. 111–122.

[6] D. Doran and S. Gokhale, "A classification framework for web robots," *Journal of the American Society of Information Science and Technology*, vol. 63, pp. 2549–2554, 2012.

[7] D. Doran, K. Morillo, and S. Gokhale, "A Comparison of Web Robot and Human Requests," in *Proc. of ACM/IEEE Conference on Advances in Social Network Analysis and Mining*, 2013, pp. 1374–1380.

[8] D. Doran, "Detection, classification, and workload analysis of web robots," Ph.D. dissertation, University of Connecticut, 2014.

[9] V. Almeida, D. A. Menascé, R. Riedi, F. P. Ribeiro, R. Fonseca, and W. Meira, Jr., "Analyzing Web robots and their impact on caching," in *Proc. Sixth Workshop on Web Caching and Content Distribution*, 2001, pp. 299–310.

[10] P. Cao and S. Irani, "Cost-aware WWW proxy caching algorithms," in *Usenix Symposium on Internet Technologies and Systems*, 1997, pp. 193–206.

[11] Q. Yang and H. H. Zhang, "Web-log mining for predictive web caching," *IEEE Transactions on Knowledge and Data Engineering*, vol. 15, no. 4, pp. 1050–1053, 2003.

[12] D. Doran and S. Gokhale, "Detecting Web Robots Using Resource Request Patterns," in *Proc. of Intl. Conference on Machine Learning and Applications*, 2012, pp. 7–12.

[13] D. Stevanovic, A. An, and N. Vlajic, "Feature evaluation for web crawler detection with data mining techniques," *Expert Systems with Applications*, vol. 39, no. 10, pp. 8707–8717, 2012.

[14] J. L. Elman, "Finding structure in time," *Cognitive science*, vol. 14, no. 2, pp. 179–211, 1990.

[15] "Weka 3: Data mining software in java," http://www.cs.waikato.ac.nz/ml/weka/, accessed: 2015-08-27.

[16] S. Karlin, *A first course in stochastic processes*. Academic Press, 1966.

[17] D. W. Hosmer Jr and S. Lemeshow, *Applied logistic regression*. John Wiley & Sons, 2004.

[18] J. R. Quinlan, *C4. 5: programs for machine learning*. Morgan Kaufmann Publishers, 1993.

[19] J. A. Suykens and J. Vandewalle, "Least squares support vector machine classifiers," *Neural information processing letters*, vol. 9, no. 3, pp. 293–300, 1999.

[20] B. Schölkopf, K.-K. Sung, C. J. Burges, F. Girosi, P. Niyogi, T. Poggio, and V. Vapnik, "Comparing support vector machines with gaussian kernels to radial basis function classifiers," *Signal Processing, IEEE Transactions on*, vol. 45, no. 11, pp. 2758–2765, 1997.

[21] A. M. Kibriya, E. Frank, B. Pfahringer, and G. Holmes, "Multinomial naive bayes for text categorization revisited," in *Advances in Artificial Intelligence*. Springer, 2005, pp. 488–499.

[22] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural networks*, vol. 2, no. 5, pp. 359–366, 1989.

[23] D. Doran and S. Gokhale, "Web robot detection techniques: Overview and limitations," *Data Mining and Knowledge Discovery*, vol. 22, no. 1, pp. 183–210, 2011.

[24] "Beautiful Soup Documentation," http://www.crummy.com/software/BeautifulSoup/bs4/doc/, accessed: 2015-08-27.

[25] M. D. Dikaiakos, A. Stassopoulou, and L. Papageorgiou, "An investigation of Web crawler behavior: characterization and metrics," *Computer Communications*, vol. 28, no. 8, pp. 880–897, 2005.