A Probabilistic Reasoning Approach for Discovering Web Crawler Sessions

Athena Stassopoulou[1] and Marios D. Dikaiakos[2]

[1] Department of Computer Science, Intercollege, Cyprus
[2] Department of Computer Science, University of Cyprus, Cyprus
stassopoulou.a@intercollege.ac.cy, mdd@cs.ucy.ac.cy

Abstract. In this paper we introduce a probabilistic-reasoning approach to detect Web robots (crawlers) from human visitors of Web sites. Our approach employs a Naive Bayes network to classify the HTTP sessions of a Web-server access log as crawler or human induced. The Bayesian network combines various pieces of evidence that were shown to distinguish between crawler and human HTTP traffic. The parameters of the Bayesian network are determined with machine learning techniques, and the resulting classification is based on the maximum posterior probability of all classes, given the available evidence. Our method is applied on real Web logs and provides a classification accuracy of 95%. The high accuracy with which our system detects crawler sessions, proves the robustness and effectiveness of the proposed methodology.

1 Introduction and Overview

In this paper, we introduce a novel approach that addresses successfully the challenging problem of automatic crawler detection using probabilistic modeling. In particular, we construct a *Bayesian network* that classifies automatically access-log sessions as being crawler- or human-induced. To this end, we combine various pieces of evidence, which, according to earlier studies [1], were shown to distinguish the navigation patterns of crawler and human user-agents of the World-Wide Web. Our approach uses machine learning to determine the parameters of our probabilistic model. The resulting classification is based on the maximum posterior probability of each class (crawler or human), given the available evidence.

To the best of our knowledge, this is one of the few published studies that propose a crawler detection system, and the only one that uses a probabilistic approach. An alternative approach that is based on decision trees, was proposed by Tan and Kumar in [7]. The authors applied their method with success on an academic access-log collected over a period of one month in year 2001.

As it will be evident from the following sections, the application of a probabilistic approach such as Bayesian Networks, is well suited for the particular domain, due to the high degree of uncertainty inherent in the problem. The Bayesian Network does not merely output a classification label, but a probability distribution over all classes by combining prior knowledge with observed data.

G. Dong et al. (Eds.): APWeb/WAIM 2007, LNCS 4505, pp. 265–272, 2007.

This probability distribution allows decisions to be made about the final classification based on how "confident" the classification is, as demonstrated by the probability distribution. For example, one need not accept weak classifications where the resulting posterior probability is less than a pre-defined minimum.

The remaining of this paper is organized as follows. In the remaining of this section, we present an overview of our approach and describe its pre-processing steps. The proposed Bayesian network classifier is introduced in Section 2. A discussion of our experiments and experimental results is given in Section 3, and we conclude in Section 4.

Overview: The goal of this work is to classify automatically an HTTP user-agent either as a crawler or a human, according to the characteristics of that agent's visit upon a Web server of interest. These characteristics are captured in the Web-server's *access logs*, which record the HTTP interactions that take place between user agents and the server. Each access-log captures a number of sessions, where each *session* is a sequence of requests issued by a single user-agent on a particular server, i.e. the "click-stream" of one user [6]. A session ends when the user completes her navigation of the corresponding site. *Session identification* is the task of dividing an access log into sessions. This is usually performed by grouping all requests that have the same IP address and using a *timeout* method to break the click-stream of a user into separate sessions [6].

Undoubtedly, there is inherent uncertainty in this approach and in any method used to identify Web sessions based on originating IP addresses. For instance, requests posted from the same IP address during the same time period do not come necessarily from the same user-agent [6]: sometimes, different user-agents may use the same IP address to access the Web (for instance, when using the same proxy server); in those cases, their activity is registered as coming from the same IP address, even though it represents different users. Also, session identification based on the heuristic timeout method carries a certain degree of uncertainty regarding the end of a user-agent's navigation inside a Web site of interest. Uncertainty in the data and the actual detection problem itself are the reasons that we believe a probabilistic approach is an ideal application to this problem.

Our system uses training to learn the parameters of a probabilistic model (Bayesian network) that classifies the user-agent of each Web session as crawler or human. To this end, the system combines evidence extracted from each Web session. Classification is based on the maximum posterior probability given the extracted evidence. The classification process comprises three main phases: (i) Access-log analysis and session identification; (ii) Learning, and (iii) classification. An overview of the functionality of our crawler-detection system is given in Algorithm 1.

2 A Bayesian Network Classifier

Feature Selection and Labeling Training Data: We base our selection of features on the characterization study of crawler behavior reported in [1].

1. Access-log analysis and session identification.
2. Session features are selected to be used as variables (nodes) in the Bayesian network.
3. Construction of the Bayesian network structure.
4. Learning:
   (a) Labeling of the set of training examples. At this step, sessions are classified as crawler- or human-initiated sessions to form the set of examples of the two classes.
   (b) Learning the required Bayesian network parameters using the set of training examples derived from step 4a.
   (c) Quantification of the Bayesian network using the learned parameters.
5. Classification: we extract the features of each session and use them as evidence to be inserted into the Bayesian network model. A probability of each session being a crawler is thus derived.

Algorithm 1. Crawler detection system

These features (attributes) are extracted for each session and provide the distinguishable characteristics between Web robots and humans. They are as follows: (i) *Maximum sustained click rate*: This feature corresponds to the maximum number of HTML requests (*clicks*) achieved within a certain time-window inside a session. The intuition behind this is that there is an upper bound on the maximum number of clicks that a human can issue within some specific time frame t, which is dictated by human factors. To capture this feature, we first set the time-frame value of t and then use a *sliding window of time* t over a given session in order to measure the maximum sustained click rate in that session. The *sliding window* approach starts from the first HTML request of a session and keeps a record of the maximum number of clicks within each *window*, sliding the window by one HTML request until we reach the last one of the given session. The maximum of all the maximum clicks per window gives the value of this attribute/feature. (ii) *Duration of session*: This is the number of seconds that have elapsed between the first and the last request. Crawler-induced sessions tend to have a much longer duration than human sessions. Human browsing behavior is more focused and goal-oriented than a Web-robot's. Moreover, there is a certain limit to the amount of time that a human can spend navigating inside a Web site. (iii) *Percentage of image requests*: This feature denotes the percentage of requests to image files (e.g. jpg, gif). The study in [1] showed that crawler requests for image resources are negligible. In contrast, human-induced sessions contain a high percentage of image requests since the majority of these image files are embedded in the Web-pages they are trying to access.(iv) *Percentage of pdf/ps requests*: This denotes the percentage requests seeking postscript(ps) and pdf files. In contrast to image requests, some crawlers, tend to have a higher percentage of pdf/ps requests than humans [1]. (v) *Percentage of 4xx error responses*: Crawlers have a higher proportion of 4xx error codes in their requests. This can be explained by the fact that human users are able to recognize, memorize and

avoid erroneous links, unavailable resources and servers [1]. (vi) *Robots.txt file request*: This feature denotes whether a request to the *robots.txt* file was made during a session. It is unlikely, that any human would check for this file, since there is no link from the Web-site to this file, nor are (most) users aware of its existence. Earlier studies showed that the majority of crawlers do not request the *robots.txt* file and so it is the *presence* of a robots.txt request in a session that will have the greater impact on it being classified as *crawler*. Therefore, a strong feature for determining the identity of a session as crawler-induced is the access to the *robots.txt*.

These features form the nodes (variables) of our Bayesian network. The Bayesian network framework enables us to combine all these pieces of evidence and derive a probability for each hypothesis (crawler vs. human) that reflects the total evidence gathered.

Our *training dataset* consists of a number of sessions, each one with its associated label (crawler or human). Since the original dataset contained thousands of sessions, it was prohibitively large to be labeled manually. Therefore, we developed a semi-automatic method for assigning labels to sessions, using heuristics. All sessions are initially assumed to be *human*. Then, we took into account a number of heuristics to label some of the sessions as crawlers: (i) IP addresses of known crawlers; (ii) The presence of HTTP requests for the Robots.txt file; (iii) Session duration values extending over a period of three hours; (iv) An HTML-to-image request ratio of more than 10 HTML files per image file.

It should be noted that we only use the first of the heuristics above to determine conclusively the label of the session as *crawler*. The other heuristics are used to give a recommended labeling of the session as *crawler*. These latter sessions are then manually inspected by a human expert to confirm or deny the suggested crawler labeling. By this semi-automatic method we aimed at minimizing the noise introduced in our training set.

Network Structure: Bayesian Networks [4] are directed acyclic graphs in which the nodes represent multi-valued variables, comprising a collection of mutually exclusive and exhaustive hypotheses. The arcs signify *direct dependencies* between the linked variables and the direction of the arcs is from *causes* to *effects*. The strengths of these dependencies are quantified by conditional probabilities. *Naive Bayes* is a special case of a Bayesian network, where a single cause (the "class") directly influences a number of effects (the "features") and the cause variable has no parents. In our proposed Bayesian network for crawler detection, each child node corresponds to one of the features presented earlier, whereas the root node represents the *class* variable. Having defined the structure of the network, we have to (i) Discretize all continuous variables; (ii) Define the conditional probability tables that quantify the arcs of the network. Subsequently, we show how we use machine learning to achieve these tasks.

Learning Network Parameters: The learning phase of the system uses the training data that have been created as described above. The training data set consists of a number of sessions, each one with its associated label (crawler or

human). For each of these sessions, we obtain the values of each of the features, described above, and which are represented as nodes in the Bayesian network. We use the data for variable quantization, based on the entropy, as well as for learning the conditional probability tables, as described in the next two sections.

Variable Quantization: Since, in this implementation, the Bayesian Network is developed for discrete variables, the continuous variables need to be quantized-divided into meaningful states (meaningful in terms of our goal, i.e. to detect crawlers). One well-known measure which characterizes the purity of the class membership of different variable states is *information content* or *entropy* [3]. The number and range of classes which result in the minimum total weighted entropy were chosen to quantize the variable. This minimum entropy principle was applied on all the continuous variables (nodes), i.e. on five out of our six features: $Clicks$, $Duration$, $Images$, PDF/PS and $Code\ 4xx$.

Conditional Probabilities: Having constructed the network nodes, we need to define the conditional probabilities which quantify the arcs of the network. More specifically, we need to define the *a priori* probability for the root node, $P(Class)$ as well as the conditional probability distributions for all non-root nodes: $P(Clicks|Class)$, $P(Duration|Class)$, $P(Images|Class)$, $P(PDF/PS|Class)$, $P(Code\ 4xx|Class)$. Each of these tables gives the conditional probability of a child node to be in each of its states, given all possible parent state combinations. We derived these probabilities from statistical data. For example, the conditional probability of $Duration$ being in class (state) 1 given $Class = Crawler$, is determined from data, by counting the number of Crawler examples with a duration within class 1, and so on.

Classification: Once the network structure is defined and the network is quantified with the learned conditional probability tables, we proceed with the classification phase of our crawler detection system. For each session to be classified, we extract the set of six features that characterize the behavior of clients and that form the variables of our Bayesian Network. As described above, the network contains only discrete variables whereas the first five of the six features are continuous-valued. Each of these feature values is therefore mapped on to a discrete state according to the ranges derived by the quantization step descrbed earlier.

Following this step, each session is now characterized by six features represented as values of discrete variables corresponding to the Bayesian network. In order to classify a session, each variable in the network is instantiated by the corresponding feature value. The Bayesian network then performs inference and derives the $belief$ in the $Class$ variable, i.e. the posterior probability of the $Class$ to take on each of its values given the evidence (features) observed. In other words we derive: $P(Class = crawler|evidence)$ and $P(Class = human|evidence)$. The maximum of the two probabilities is the final classification given to the session.

3 Experimental Results

In this section we present the experiments performed in order to apply our methodology and evaluate the performance of our crawler detection system.

Training Data sets: For the purposes of evaluating the performance of our crawler detection system, we obtained access logs from two servers of two academic institutions: the University of Toronto and the University of Cyprus. The access logs were processed by our log analyzer to extract the sessions. These sessions, the majority being from the University of Toronto, were used for training. Sessions were then labeled using our approach described earlier. The learning stage proved to be challenging task. The problem encountered with this stage is one of *class imbalance* [5]. The data sets present a class imbalance when there are many more examples of one class than of the other. It is usually the case that this latter class, i.e. the unusual class, is the one that people are interested in detecting. Because the unusual class is rare among the general population, the class distributions are very skewed [5]. The study reported in [1] have concluded that crawler activity in access logs amount to less than 10 per cent of the total number of requests. To tackle the problem of imbalanced data sets we used *resampling* and adopted two resampling approaches: random oversampling and random undersampling. We performed 5 experiments, based on resampling (both oversampling and undersampling) at various ratios.

Table 1 shows the number of Crawler and Human sessions in each of the 5 training data sets created via resampling. The last column shows the prior probability distributions of variable *Class*, considering the distribution of sessions actually used for training.

We constructed five Bayesian network classifiers, one for each experiment. The networks had the same structure but differed in their parameters, i.e. prior probabilities, conditional probability tables and quantization ranges. Each time a new training data set was introduced, new network parameters were derived using training on the new set.

Testing the system: A different access log, from the ones not used during training, was randomly chosen for testing. Since the majority of the sessions used for training were extracted from the University of Toronto log, we have chosen a different institution server altogether to evaluate our detection system. This access log used for testing was obtained from the University of Cyprus and spanned a

Table 1. Data sets used for five experiments with and without resampling

Data Set No.	No. Distinct Humans	No. Distinct Crawlers	No. Humans used in training	No. Crawlers used in training	Prior Probabilities: (Human, Crawler)
1	10106	988	10106	988	(0.91, 0.09)
2	10106	988	10106	1784	(0.85, 0.15)
3	10106	988	10106	10106	(0.5, 0.5)
4	10106	988	5599	988	(0.85, 0.15)
5	10106	988	988	988	(0.5, 0.5)

Table 2. Evaluation metrics of each Bayesian network classifier

Classifier	Recall	Precision	$F_1 - measure$
C1	0.80	0.92	0.855
C2	0.81	0.93	0.866
C3	0.95	0.86	0.903
C4	0.81	0.93	0.866
C5	0.95	0.79	0.863

period of one month. A human expert did an entirely manual classification of each session, extracted by our log analyzer from this the testing set, in order to provide us with the ground truth by which we were to evaluate our classifier's performance. It should be noted that we did not do any resampling for the testing.

We tested the performance of all five Bayesian networks (one for each data set), on the same testing dataset[1]. The testing set contained 685 actual human sessions and 99 actual crawler sessions, as labeled by an independent human expert. Throughout this section we will refer to the 5 classifiers as follows: (i) Classifier $C1$: Obtained using learning of Data set 1 (no resampling); (ii) Classifier $C2$: Obtained using learning of Data set 2 (oversampling to 15%); (iii) Classifier $C3$: Obtained using learning of Data set 3 (oversampling to 50%-equally represented classes); (iv) Classifier $C4$: Obtained using learning of Data set 4 (undersampling to 85%); (v) Classifier $C5$: Obtained using learning of Data set 5 (undersampling to 50%-equally represented classes).

Two metrics that are commonly applied to imbalanced datasets to evaluate the performance of classifiers is *recall* and *precision*. These two metrics are summarized into a third metric known as the F_1-measure [8]. The values of recall, precision and F_1-measure obtained by classifiers $C1, \ldots, C5$ are given in Table 2.

As it can be seen from table 2, our crawler detection system yields promising results with both recall and precision being above 79% in all experiments performed. The lowest $F1$-measure is obtained by C1 when we train the system with the dataset without resampling. The prior probability of a session to be *Human* in that dataset was 91% and the classifier was therefore biased towards humans. It missed only 7 out of the 685 *Human* sessions but sacrificed recall, by missing 20 out of the 99 actual *Crawler* sessions. By resampling so that the *Crawler* class amounts to 85% of the sessions (either via oversampling as in C2 or by undesampling as in C4) we have slightly improved results compared to C1. Both C2 and C4 have the same precision and recall. The best results are obtained by C3, which was trained using oversampling of *Crawlers* so that they reach the number of *Human* examples in the original set. The recall, i.e. the percentage of crawlers correctly classified increases dramatically to 95%, with 94 sessions correctly classified as *Crawlers* out of 99 actual crawlers. This causes a decrease in precision, which is nevertheless not so dramatic. The same recall as C3 is achieved by C5 which was trained by undersampling *Humans* so that both classes are again, equally represented. However, this caused a significant decrease in precision to 79%, i.e. we have

[1] The networks were implemented using the $Ergo^{TM}$ tool [2].

an increase in the number of false positives, i.e. *Humans* incorrectly classified as *Crawlers*. The significant decrease in precision of C5, is not surprising since, with random undersampling there is no control over which examples are eliminated from the original set. Therefore significant information about the decision boundary between the two classes may be lost. The risk with random oversampling is to do overfitting due to placing exact duplicates of minority examples from the original set and thus making the classifier biased by "remembering" examples that were seen many times. The are other alternatives to random resampling which may reduce the risks outlined above. An investigation and a comparison of the various resampling techniques is beyond the scope of the current paper.

4 Conclusion

In this paper we have presented the use a Bayesian network, for detecting Web crawlers from access logs. This Bayesian approach is well suited for the particular domain due to the high degree of uncertainty inherent in the problem. Our system uses machine learning to determine the parameters of the Bayesian network that classifies the user-agent of each Web session as crawler or human. The system combines evidence extracted from each Web session to determine the class it belongs to. The Bayesian network does not merely output a classification label, but a probability distribution over all classes by combining prior knowledge with observed data. We have used resampling to counter the class imbalance problem and developed five classifiers by training on five different datasets. The high accuracy with which our system detects crawler sessions, proves the effectiveness of our proposed methodology.

References

1. M. D. Dikaiakos, A. Stassopoulou, and L. Papageorgiou. An Investigation of WWW Crawler behavior: Characterization and Metrics. *Computer Communications*, 28(8):880–897, May 2005.
2. Noetic Systems Incorporated. http://www.noeticsystems.com/ergo/index.shtml.
3. T. M. Mitchell. *Machine Learning*. McGraw Hill Companies Inc., 1997.
4. J. Pearl. *Probabilistic reasoning in intelligent systems: Networks of plausible inference*. Morgan Kaufmann Publishers Inc., 1988.
5. F. J. Provost and T. Fawcett. Analysis and visualization of classifier performance: Comparison under imprecise class and cost distributions. In *Proceedings of the Third International Conference on Knowledge Discovery and Data Mining*, pages 43–48, 1997.
6. J. Srivastava, R. Cooley, M. Deshpande, and P.-N. Tan. Web Usage Mining: Discovery and Applications of Usage Patterns from Web Data. *SIGKDD Explorations*, 1(2):12–23, January 2000.
7. P.-N. Tan and V. Kumar. Discovery of Web Robot Sessions Based on their Navigational Patterns. *Data Mining and Knowledge Discovery*, 6(1):9–35, January 2002.
8. P.-N. Tan, M. Steinbach, and V. Kumar. *Introduction to Data Mining*. Addison-Wesley, 2005.