

Marked Event–Time 5-Grams for Bot Detection

Hong Chen^{1*}

^{1*}Indeed.

Corresponding author(s). E-mail(s): hcen@indeed.com;

Abstract

Malicious bots can scrape online content, abuse services designed for human users, and distort business metrics. Bot detection is therefore an important task for many online services. We propose a supervised marked 5-gram bot detector that represents a client’s HTTP behavior using three consecutive request types and the two intervening time gaps. Given a client’s request history, it outputs one log-likelihood-ratio score to support history-level bot classification. We evaluated the detector on about 50,000 pseudonymous client histories collected during an Indeed HTTP observation window longer than 12 hours, using fivefold out-of-fold scoring and comparing it with a marked first-order adaptation of Doran and Gokhale’s published Markov detector (the first-order comparator). The marked 5-gram achieved a history-level area under the receiver operating characteristic curve (ROC-AUC) of 0.7955 and average precision of 0.5666, compared with 0.7149 and 0.4827 for the first-order comparator; both methods assigned a finite score to every history. The marked 5-gram’s ROC-AUC was 0.0806 higher.

Keywords: web bot detection, marked n-grams, likelihood-ratio scoring, Markov models, behavioral sequence modeling

1 Introduction

Internet services depend on automated clients for search indexing, monitoring, accessibility, and partner integrations. Malicious bots, however, can also issue requests at scale to scrape content, exhaust resources, probe accounts, and manipulate traffic, while distorting business metrics such as daily active user counts. Bot detection is therefore operationally important.

On the server side, request logs provide two useful behavioral signals for each HTTP client: which resources it requests and how quickly it issues successive requests. In practice, these logs are typically grouped by client and partitioned into sessions

using a predefined inactivity threshold before bot analysis. Heuristic bot detectors based on counts, average timings, or click velocity discard information carried by request order and can therefore overlook subtler bot behavior. Doran and Gokhale’s novel first-order Markov detector [1] factors sessions into adjacent-transition terms and supports both retrospective and real-time applications. We investigate whether additionally parameterizing joint three-request/two-gap motifs yields higher history-level ranking metrics.

Here, a *history* is the chronologically ordered sequence of requests associated with one pseudonymous client identifier within an observation window longer than 12 hours. Unlike a session, a history is not split at an inactivity threshold, thereby preserving request-order and timing information across potential session boundaries.

We propose a supervised, class-conditional marked 5-gram model. For example, for request types A, B, C, D separated by short (s), medium (m), and long (l) gaps, the history yields two overlapping patterns, (A, s, B, m, C) and (B, m, C, l, D) . “Marked” means that timing categories accompany the request types; “class-conditional” means that separate pattern distributions are learned for the two classes, bot=1 (bot-labeled) and bot=0 (not-bot-labeled). To score a new request history, the model looks up the probability of every overlapping pattern in both distributions and averages the resulting log probability ratios. Patterns more common in bot=1 histories contribute positive evidence, whereas patterns more common in bot=0 histories contribute negative evidence. The output is one continuous score for the complete history, not a decision about an individual request or 5-gram.

We benchmark this model against a marked adaptation of Doran and Gokhale’s published first-order detector [1] (the first-order comparator), using the same histories, event representation, timing bins, training folds, and smoothing. Bot labels derived from scores supplied by an external vendor serve as the benchmark ground truth for model training and evaluation in this study. The marked 5-gram’s pooled area under the receiver operating characteristic curve (ROC-AUC) was 0.0806 higher and its average precision was 0.0839 higher, with a finite score for every history.

2 Related Work

Server-side automation detection has long used behavioral evidence from request sessions. Tan and Kumar represent sessions with navigational features and a decision tree [2], Stassopoulou and Dikaiakos use probabilistic reasoning over session evidence [3], and Xu et al. combine abnormal labels derived by expectation-maximization (EM) with decision-tree rules at the IP level in a large e-commerce marketplace [4]. These studies establish the value of server-log behavior but rely mainly on engineered features.

Timing also provides behavioral evidence in automation and click-fraud settings. Gianvecchio et al. characterize human and automated Internet-chat behavior using first-order entropy and corrected conditional entropy of message interarrival times and message sizes [5]. Nagaraja and Shah’s Clicktok pipeline detects click spam by

exploiting reuse and correlation in click timing across sources [6]. These studies motivate retaining gap information alongside event identity, although our method models local event-gap motifs rather than entropy or aggregate traffic decompositions.

Joint event-and-time sequence representations most directly motivate our proposed representation and scoring rule. Cresci et al. encode online action sequences as digital DNA and use common subsequences to distinguish spambot groups from genuine accounts [7]. Mei and Eisner jointly model event types and event times in a generative temporal process [8]. Most directly, Wang et al.’s **Hybrid/5gram+count** representation interleaves three consecutive click types with the two discretized interarrival gaps between them for clustering-based Sybil detection [9]; we retain this three-event/two-gap construction for history-level bot ranking. Suchacka et al.’s neural sequential detector (NNSEQ) [10] supplies a second scoring precedent by accumulating per-request class log-ratio evidence, although it uses a neural estimator and thresholded sequential decisions.

For benchmarking, we adapted the first-order comparator from Doran and Gokhale’s published web-robot detector [1]. Their method fits separate human and robot first-order discrete-time Markov chains (DTMCs) over resource types, including a start distribution and transition matrix, and compares full-session or prefix log likelihoods. We preserve the class-conditional start/transition log-likelihood-ratio evidence but hold the rest of the pipeline fixed to enable the benchmark.

3 Marked 5-Gram Likelihood-Ratio Score

3.1 Representation

Consider the four-request history

$$A \xrightarrow{s} B \xrightarrow{m} C \xrightarrow{l} D,$$

where s, m, l denote short, medium, and long time gaps. The model extracts two overlapping marked 5-grams:

$$(A, s, B, m, C), \quad (B, m, C, l, D).$$

During training, each history is converted to normalized frequencies of such patterns. Separate smoothed tables are learned for bot-labeled and not-bot-labeled histories. For a new history, every pattern contributes the log of its probability in the bot-labeled table divided by its probability in the not-bot-labeled table. For example, hypothetical probabilities 0.04 and 0.01 contribute $\log(4) \approx 1.39$; equal probabilities contribute zero. The average contribution is the score for the complete history.

Formally, let history c contain $N_c \geq 3$ chronologically ordered HTTP requests, and let t_{ci} be the timestamp of request i . After the fold-local preprocessing described in Section 4.3, $V_{ci} \in \mathcal{V} = \{0, \dots, 4\}$ is the event index for request i : the four most frequent learned request categories plus an **OTHER** category. For $i \geq 2$, $Z_{ci} \in \mathcal{Z} = \{0, 1, 2\}$ is the learned bin containing the raw gap $t_{ci} - t_{c,i-1}$. Let $\mathcal{W} = \mathcal{V} \times \mathcal{Z} \times \mathcal{V} \times \mathcal{Z} \times \mathcal{V}$. For every $i = 3, \dots, N_c$, define

$$W_{ci} = (V_{c,i-2}, Z_{c,i-1}, V_{c,i-1}, Z_{ci}, V_{ci}) \in \mathcal{W}. \quad (1)$$

Here “5-gram” means five alternating coordinates—three events and two gaps—not five requests. The Cartesian state space has

$$|\mathcal{W}| = 5 \times 3 \times 5 \times 3 \times 5 = 1,125 \quad (2)$$

possible cells. Windows overlap but never cross a history boundary, and the implementation adds no boundary or padding token. Each window therefore preserves a two-transition motif that a first-order table separates into individual transitions.

3.2 Equal-history class tables

For cell $w \in \mathcal{W}$, the normalized within-history frequency [9] is

$$f_{cw} = \frac{\text{count}_c(w)}{N_c - 2}, \quad \sum_{w \in \mathcal{W}} f_{cw} = 1. \quad (3)$$

Let $y = 1$ denote the bot-labeled class and $y = 0$ the not-bot-labeled class, and let C_y be the training histories with label y . We sum normalized history vectors rather than raw windows:

$$S_{yw} = \sum_{c \in C_y} f_{cw}. \quad (4)$$

Consequently, a history containing 1,000 requests and one containing 10 requests each contribute total mass one to its class table. Long histories therefore do not mechanically dominate the class-table sufficient statistics; path-frequency counts, event-vocabulary fitting, and gap-bin fitting remain request/gap weighted.

With fixed additive smoothing $\alpha = 0.5$ [11],

$$P_y(w) = \frac{S_{yw} + 0.5}{|C_y| + 0.5(1,125)}. \quad (5)$$

Smoothing assigns positive probability to every cell, including one unseen in a training class. No class-prior odds term is added. All logarithms are natural.

For a deliberately small example, consider one 5-gram cell w . Suppose two bot-labeled training histories contain seven and eight windows, and w occurs seven and four times. Equation (3) gives within-history frequencies $7/7 = 1$ and $4/8 = 0.5$. Suppose two not-bot-labeled histories contain eight and seven windows, and w occurs four and zero times, giving frequencies $4/8 = 0.5$ and $0/7 = 0$. Equation (4) therefore gives

$$S_{1w} = 1 + 0.5 = 1.5, \quad S_{0w} = 0.5 + 0 = 0.5.$$

Because each class contains two training histories, the denominator in Equation (5) is $2 + 0.5(1,125) = 564.5$, and

$$P_1(w) = \frac{1.5 + 0.5}{564.5} = 0.003543, \quad P_0(w) = \frac{0.5 + 0.5}{564.5} = 0.001771.$$

Thus this pattern is twice as probable under the bot-labeled table, and its local log-likelihood-ratio contribution in Section 3.3 is $\log[P_1(w)/P_0(w)] = \log(2) = 0.693$. This is positive evidence favoring the bot-labeled class, not a 69.3% bot probability. In particular, $P_1(w)$ denotes $P(W = w \mid Y = 1)$, not $P(Y = 1 \mid W = w)$.

3.3 Scoring

The marked 5-gram score is a class-conditional log-likelihood ratio [1]:

$$L_c = \sum_{w \in \mathcal{W}} f_{cw} \log \frac{P_1(w)}{P_0(w)}. \quad (6)$$

Continuing the example from Section 3.2, consider a new unlabeled history c containing ten requests and therefore $N_c - 2 = 8$ marked 5-gram windows. Suppose four windows are the previously defined cell w , for which $P_1(w)/P_0(w) = 2$, and the other four are a cell v having equal probability in the two class tables. Thus $f_{cw} = 4/8 = 0.5$, $f_{cv} = 4/8 = 0.5$, and $P_1(v)/P_0(v) = 1$. Only these two cells have nonzero frequency, so Equation (6) gives $L_c = 0.5 \log(2) + 0.5 \log(1) = 0.5(0.693) + 0.5(0) \approx 0.347$.

The positive score means that the history’s average local evidence favors the bot-labeled class. The value 0.347 is a history-level mean LLR, not a 34.7% bot probability or a binary decision.

Equivalently, L_c is the average local log evidence favoring the bot-labeled class over the not-bot-labeled class across the $N_c - 2$ windows. Positive values favor bot-labeled histories by construction. Individual 5-grams supply evidence, but the model returns one score for the complete history. Because windows overlap, this is a mean local log-likelihood-ratio (LLR) score, not a calibrated posterior probability or a claim that the windows are independent. The model does not choose a decision threshold.

4 Benchmark Design

4.1 Doran-lineage first-order comparator

Doran and Gokhale [1] group requests into IP/user-agent sessions, reduce resources to nine hand-built classes, and fit one start vector and one 9×9 event-transition matrix per class. Their offline detector compares full-session log likelihoods; their real-time variant accumulates a prefix LLR after a minimum number of requests and decides when its magnitude crosses a threshold.

Like Doran and Gokhale’s detector, the first-order comparator (1) learns a separate start distribution and transition model for each class, (2) conditions each transition on only the immediately preceding event, (3) scores by comparing log likelihoods under the two class models, and (4) analytically assigns positive scores to bot-favoring evidence without a learned sign. Table 1 records the consequential changes.

For history c , with $a, b \in \mathcal{V}$ and $z \in \mathcal{Z}$, define one start indicator and normalized marked-transition frequencies:

$$u_c(a) = \mathbf{1}\{V_{c1} = a\}, \quad (7)$$

$$g_c(a, z, b) = \frac{1}{N_c - 1} \sum_{i=2}^{N_c} \mathbf{1}\{V_{c,i-1} = a, Z_{ci} = z, V_{ci} = b\}. \quad (8)$$

Both u_c and g_c sum to one over their respective axes. Within outer-training class C_y , let

Table 1 Comparison of Doran and Gokhale’s published detector with our benchmark adaptation (the first-order comparator).

Design element	Doran and Gokhale [1]	Indeed benchmark adaptation
Unit	IP + user-agent session; adjacent gaps below 30 min	One observation-window history per pseudonymous client identifier
Event states	Nine URI/resource classes	Four fold-fitted path symbols + OTHER
Time	Session boundary only	Three fold-fitted gap bins mark each transition
Kernel	$P(\text{next event} \mid \text{previous event})$	$P(\text{gap, next event} \mid \text{previous event})$
Training weight	Raw pooled transitions	One start unit and one normalized transition-mass unit per history
Zero cells	No smoothing reported	Fixed additive pseudocount 0.5
Score	Unnormalized sequence/prefix LLR	Full-history mean LLR
Evaluation	Hard offline or thresholded early decision	Continuous out-of-fold ROC-AUC and average precision

$$U_y(a) = \sum_{c \in C_y} u_c(a), \quad G_y(a, z, b) = \sum_{c \in C_y} g_c(a, z, b), \quad (9)$$

and $G_y(a) = \sum_{z,b} G_y(a, z, b)$. The smoothed start vector and transition tensor are

$$Q_y(a) = \frac{U_y(a) + 0.5}{|C_y| + 0.5(5)}, \quad (10)$$

$$T_y(z, b \mid a) = \frac{G_y(a, z, b) + 0.5}{G_y(a) + 0.5(15)}. \quad (11)$$

The first-order comparator stores 5 start cells and $5 \times 3 \times 5 = 75$ transition cells per class, versus 1,125 marked 5-gram cells. Its score is

$$M_c = \frac{1}{N_c} \left[\log \frac{Q_1(V_{c1})}{Q_0(V_{c1})} + \sum_{i=2}^{N_c} \log \frac{T_1(Z_{ci}, V_{ci} \mid V_{c,i-1})}{T_0(Z_{ci}, V_{ci} \mid V_{c,i-1})} \right]. \quad (12)$$

The bracketed sum contains one start term and $N_c - 1$ transition terms, so division by N_c produces mean evidence per modeled token. Although “DTMC” identifies its research lineage, the joint gap/next-event outcome makes the first-order comparator a marked transition model.

4.2 Indeed data and bot labels derived from Cloudflare scores

We analyzed cleaned Indeed HTTP requests from a single observation window longer than 12 hours. Records contained a request time, path, pseudonymous client-history identifier, and Cloudflare bot score. After exact-row deduplication and six deterministic path-cleaning rules, including removal of static assets, requests were ordered into one observation-window history per identifier.

Each request cast an analyst-defined vote: scores 1–14 voted bot, whereas missing scores, score 0, and scores 15–99 voted not bot. A majority vote determined the bot

label of each resolved history. Cloudflare describes score 1 as automated, scores 2–29 as likely automated, scores 30–99 as likely human, and score 0 as unevaluated; missing scores carry no vendor assertion.

Table 2 Dataset and cohorts from a single Indeed HTTP observation window longer than 12 hours.

Quantity	Value
Cleaned HTTP requests	About 4.6 million
Pseudonymous client histories	About 1.3 million
Histories with a resolved bot label	More than 99% of histories
Primary histories with at least 9 requests	About 50,000
Primary bot-labeled prevalence	About 22%

The primary cohort required at least nine requests, providing at least seven marked 5-grams and excluding roughly 96% of resolved histories. All performance metrics use the full, unrounded cohorts.

4.3 Fold-local cross-fitted pipeline

For the primary cohort, we used stratified fivefold cross-validation grouped by pseudonymous client-history identifier, keeping each complete history in one fold. For each held-out fold, all preprocessing and both models were learned from the other four folds. Training paths occurring fewer than 30 times and unseen held-out paths were mapped to `<all-other-path>`. After normalization, the four most frequent training symbols remained distinct; all others were mapped to `OTHER`. The one-third and two-thirds quantiles of raw, unclipped training gaps defined three timing bins, with boundary values entering the higher bin. Both models therefore received the same fold-local event and gap sequences and produced one out-of-fold score per history.

We evaluated complete histories on the models’ common finite-score support using ROC-AUC, average precision (AP), and finite coverage. ROC-AUC measures ranking; AP summarizes precision–recall and should be read against the roughly 22% bot-label prevalence; finite coverage is the share receiving a numeric score. Because we selected no operating threshold, we report neither accuracy nor false-positive rate.

5 Results

5.1 Primary ranking performance

The marked 5-gram scored higher on both metrics: ROC-AUC 0.7955 versus 0.7149 and average precision 0.5666 versus 0.4827, absolute differences of 0.0806 and 0.0839. Both models produced one finite score for every one of the roughly 50,000 held-out histories, so their common coverage was 100% (Table 3).

Table 3 Pooled fivefold out-of-fold performance on the primary cohort of about 50,000 histories with about 22% bot-labeled prevalence. Average precision should be read relative to that prevalence; finite coverage is the share of histories with a numeric score.

Method	ROC-AUC	Average precision	Finite coverage
Marked 5-gram LLR	0.7955	0.5666	100%
First-order marked transition LLR	0.7149	0.4827	100%
Marked 5-gram minus first-order marked transition	+0.0806	+0.0839	—

5.2 Sensitivity and diagnostics

The longer-history sensitivity analysis independently reran the complete fivefold pre-processing and model-fitting pipeline on histories containing at least 30 requests ($N_c \geq 30$), which provide at least 28 marked 5-grams. Table 4 compares it with the primary cohort.

Table 4 Pooled fivefold out-of-fold performance for the primary (at least 9 requests) and longer-history (at least 30 requests) cohorts from the same observation window. Counts and prevalence are rounded; metrics use the unrounded cohorts.

History subset	Histories	Bot-labeled prevalence	Marked 5-gram ROC-AUC / average precision	First-order comparator ROC-AUC / average precision	ROC-AUC difference (5-gram minus first-order)
At least 9 requests	About 50,000	About 22%	0.7955 / 0.5666	0.7149 / 0.4827	+0.0806
At least 30 requests	About 9,000	About 19%	0.7804 / 0.5063	0.5733 / 0.4024	+0.2071

Both methods had 100% finite coverage in the longer-history subset. This subset has a different client composition and lower bot-labeled prevalence (about 19%), so its larger observed separation is descriptive and cannot replace or strengthen the primary evidence.

Table 5 shows a moderate positive correlation between the score rankings (Spearman $\rho = 0.3929$) and that neither score had an absolute rank correlation above 0.1 with request count.

5.3 Limitations

Three limitations bound these results. First, although the vendor-derived labels serve as the benchmark ground truth, they are analyst-defined proxies rather than adjudicated bot identities. ROC-AUC therefore measures agreement with the voting rule. Deploying the models in production would likely require a series of training sets derived from human-labeled historical traffic.

Table 5 Diagnostics for the primary cohort of histories with at least nine requests. Spearman correlations use pooled raw out-of-fold scores. An occupied cell has positive unsmoothed training mass; occupied-cell values are means across the five outer-training folds and are descriptive.

Diagnostic	Value	Interpretation
Rank correlation between model scores	0.3929	The score rankings have a moderate positive association.
Score correlation with request count: 5-gram / first-order	−0.0086/0.0703	Both absolute correlations are below 0.1.
Mean per-history share of absolute local first-order evidence from the start term	6.9%	Most absolute first-order evidence came from transitions.
Occupied 5-gram cells: bot=0 / bot=1	956.8 / 569.2 of 1,125	The bot-labeled class occupied fewer distinct pattern cells in training.
Occupied first-order transitions: bot=0 / bot=1	75.0 / 72.8 of 75	Nearly all transition cells appeared in both classes.

Second, the models are not capacity- or factorization-matched. The marked 5-gram has 1,125 cells per class versus 80 for the comparator, and they differ in start treatment, probability factorization, and length normalization. Its advantage therefore cannot be attributed to local context alone.

Third, random folds within one observation window test same-period interpolation, not transfer across time, platforms, or traffic regimes. Related histories may remain dependent despite identifier-level grouping, and an observation-window history need not be a natural session.

Data and Code Availability. Raw Indeed HTTP requests, pseudonymous client identifiers, request paths, per-history outputs, and learned probability tables are restricted and cannot be released publicly. If organizational approval is obtained, reproducibility materials would include the benchmark configuration, evaluator, aggregate tables, and a non-sensitive synthetic example.

References

- [1] Doran, D., Gokhale, S.S.: An integrated method for real time and offline web robot detection. *Expert Systems* **33**(6), 592–606 (2016) <https://doi.org/10.1111/exsy.12184>
- [2] Tan, P.-N., Kumar, V.: Discovery of web robot sessions based on their navigational patterns. *Data Mining and Knowledge Discovery* **6**(1), 9–35 (2002) <https://doi.org/10.1023/A:1013228602957>
- [3] Stassopoulou, A., Dikaiakos, M.D.: Web robot detection: A probabilistic reasoning approach. *Computer Networks* **53**(3), 265–278 (2009) <https://doi.org/10.1016/j.comnet.2008.09.021>
- [4] Xu, H., Li, Z., Chu, C., Chen, Y., Yang, Y., Lu, H., Wang, H., Stavrou, A.:

- Detecting and characterizing web bot traffic in a large e-commerce marketplace. In: ESORICS 2018. Lecture Notes in Computer Science, vol. 11099, pp. 143–163. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-98989-1_8
- [5] Gianvecchio, S., Xie, M., Wu, Z., Wang, H.: Humans and bots in internet chat: Measurement, analysis, and automated classification. *IEEE/ACM Transactions on Networking* **19**(5), 1557–1571 (2011) <https://doi.org/10.1109/TNET.2011.2126591>
 - [6] Nagaraja, S., Shah, R.: Clicktok: Click fraud detection using traffic analysis. In: *Proceedings of the 12th Conference on Security and Privacy in Wireless and Mobile Networks*, pp. 105–116. ACM, New York, NY (2019). <https://doi.org/10.1145/3317549.3323407>
 - [7] Cresci, S., Di Pietro, R., Petrocchi, M., Spognardi, A., Tesconi, M.: DNA-inspired online behavioral modeling and its application to spambot detection. *IEEE Intelligent Systems* **31**(5), 58–64 (2016) <https://doi.org/10.1109/MIS.2016.29>
 - [8] Mei, H., Eisner, J.M.: The neural hawkes process: A neurally self-modulating multivariate point process. In: *Advances in Neural Information Processing Systems* 30, pp. 6754–6764 (2017)
 - [9] Wang, G., Konolige, T., Wilson, C., Wang, X., Zheng, H., Zhao, B.Y.: You are how you click: Clickstream analysis for sybil detection. In: *22nd USENIX Security Symposium*, pp. 241–256 (2013). <https://www.usenix.org/conference/usenixsecurity13/technical-sessions/presentation/wang>
 - [10] Suchacka, G., Cabri, A., Rovetta, S., Masulli, F.: Efficient on-the-fly web bot detection. *Knowledge-Based Systems* **223**, 107074 (2021) <https://doi.org/10.1016/j.knosys.2021.107074>
 - [11] Nemenman, I., Shafee, F., Bialek, W.: Entropy and inference, revisited. In: *Advances in Neural Information Processing Systems* 14, pp. 471–478 (2002). <https://doi.org/10.7551/mitpress/1120.003.0065>