AdvCat: Domain-Agnostic Robustness Assessment for Cybersecurity-Critical Applications with Categorical Inputs

Helene Orsini, Hongyan Bao, Yujun Zhou, Xiangrui Xu, Yufei Han, Longyang Yi, Wei Wang, Xin Gao, Xiangliang Zhang

AdvCat: Domain-Agnostic Robustness Assessment for Cybersecurity-Critical Applications with Categorical Inputs

Helene Orsini*†, Hongyan Bao*‡, Yujun Zhou*‡, Xiangrui Xu*§,
Yufei Han†, Longyang Yi§, Wei Wang§, Xin Gao‡, Xiangliang Zhang‡¶
†INRIA, France ‡King Abdullah University of Science and Technology, Thuwal, SA
§Beijing Jiaotong University, Beijing, CN
¶University of Notre Dame, Indiana, USA
helene.orsini@irisa.fr, {hongyan.bao,yujun.zhou}@kaust.edu.sa, xiangrui.xu@bjtu.edu.cn,
yufei.han@inria.fr, {longyang.yi,wangwei1}@bjtu.edu.cn, xin.gao@kaust.edu.sa, xzhang33@nd.edu

***Abstract*—Machine Learning-as-a-Service systems (MLaaS) have been largely developed for cybersecurity-critical applications, such as detecting network intrusions and fake news campaigns. Despite effectiveness, their robustness against adversarial attacks is one of the key trust concerns for MLaaS deployment. We are thus motivated to assess the adversarial robustness of the Machine Learning models residing at the core of these security-critical applications with categorical inputs. Previous research efforts on accessing model robustness against manipulation of categorical inputs are specific to use cases and heavily depend on domain knowledge, or require white-box access to the target ML model. Such limitations prevent the robustness assessment from being as a domain-agnostic service provided to various real-world applications. We propose a provably optimal yet computationally highly efficient adversarial robustness assessment protocol for a wide band of ML-driven cybersecurity-critical applications. We demonstrate the use of the domain-agnostic robustness assessment method with substantial experimental study on fake news detection and intrusion detection problems.**

***Index Terms*—Adversarial robustness assessment, cybersecurity application, categorical inputs, intrusion detection, misinformation detection**

I. Introduction

Witnessed during recent decades, Machine Learning (ML)-based techniques have powered increasingly more cybersecurity-critical services, such as intrusion detection and online misinformation investigation. The ML-based decision output in these applications directly impact human security analysts' judgment and strategic actions, like preventing underlying high-stake cyber intrusions and poisoning fake news [1] spreading in the early stage. However, a seemingly accurate ML-based detector/classifier may be highly vulnerable to adversarial input perturbations, which result in misdetection of security incidents. The lack of robustness against such input perturbations raises trust concerns over the use of ML models in cybersecurity-critical applications [2]. Therefore, to provide trustable ML-as-a-Service in these applications, it has become a must to assess the adversarial robustness of the core ML models, beyond achieving accurate detection/classification. *The robustness assessment discloses to what extent the performance of a core ML model can be affected by adversarial attacks at what expenses*.

There are three unaddressed key challenges in approaching the adversarial robustness assessment target in these cybersecurity-critical scenarios. **First**, the previous gradient-based approaches on understanding the robustness of deep learning models focus on continuous input [3]. They are ineffective and infeasible for the ML models with non-numerical categorical inputs. However, those categorical inputs pervasively exist in cybersecurity-critical applications [4], [5]. For instance, intrusion detection identifies cyber attacks via a rich set of categorical behavioral signatures, such as malware execution traces, malicious network communication logs, and system event logs [5], [6]. Misinformation detection is often based on learning the relationship of various grammatical categories such as word co-occurrences, subject-predicate-object triples, and writing styles [1], [4]. Unlike continuous measurements such as pixel intensities, categorical inputs have a limited number of options for the category values and have no intrinsic ordering to the categories. Conducting adversarial perturbations on categorical data are in nature an *NP-hard knapsack problem* [7]. As such, it is difficult to define a computationally efficient strategy to produce adversarial perturbations over categorical inputs.

Second, heuristic rules or trial-and-error processes may lead to feasible perturbations with additional assumptions narrowing the combinatorial perturbation space [8]. These methods nonetheless depend heavily on domain-specific knowledge and constraints defined by semantic integrity and rules. Such dependency limits the adaptive potential of the domain-specific assessment protocols to new domains. Moreover, domain-specific knowledge may not be always readily available. For example, the threat settings of cyber attacks vary drastically across different attack techniques and IT system architectures [5], [6]. Encoding domain-specific contexts of various intrusion incidents require expensive investigation overheads

*The first four authors contributed equally.
¶Corresponding author.

on a case-by-case basis. Besides, system threats may stay unknown to security analysts when an attack is delivered. It is impossible to define domain-specific rules for the zero-day attack events. The absence of *a principled and domain-agnostic robustness assessment protocol* makes it difficult to provide an assessment-as-a-service pipeline to measure the adversarial vulnerability of different cybersecurity-critical applications.

Third, providing robustness assessment-as-a-service requires *blackbox access* to the target ML model for safeguading the intellectual property. However, most existing attack methods require computing the gradient-based estimator to seek optimal adversarial data manipulations [7]–[9], and thus require a *white-box access* to the target ML model. Even though a few black-box adversarial attack methods [10]–[13] have been proposed, they either work on continuous data, or they are tailored to specific applications and can be barely extended across different domains.

We are thus motivated to address these challenges and build **an out-of-the-box framework of Robustness Assessment-as-a-Service (RAaaS) on categorical inputs**. This framework presents an **automated and generally applicable robustness assessment pipeline** for any cybersecurity-critical domain with categorical inputs, even without significant domain-specific expertise. The produced diagnostic reports help benchmark the utility of the ML-assisted detection/classification under adversarial perturbations.

Our framework, named ***AdvCat***, employs three model-agnostic methods to solve the NP-hard knapsack problem of the robustness assessment task regarding categorical inputs, Forward Stepwise Greedy Search (*FSGS*), Stochastic Greedy Search (*SGS*), and Upper-Confidence Bound Search (*UCBS*) method. They conduct an iterative combinatorial search of feasible perturbations over categorical features, based on the classification output queried from the ML model. The diagnostic report generated by *AdvCat* covers the performance change (e.g., drop of accuracy) caused by adversarial perturbation and the expense of the robustness assessment process (e.g., the number of queries and runtime cost). We highlight below the advantages making *AdvCat* a unique and valuable asset to the community and summarize our contributions:

- *AdvCat*, by design, is domain-agnostic and generally usable for any application with categorical inputs. However, if necessary, *AdvCat* can be further customized to specific applications by integrating additional domain-dependent rules as plugins. Besides, *AdvCat* provides computationally efficient and provably accurate robustness assessment, which consolidates the use of *AdvCat* across various ML models and application scenarios.
- *AdvCat* only needs to query the decision output of the target ML model over a set of anonymized testing instances provided by the ML-driven application owner. This black-box setting safeguards data privacy and the intellectual property of ML models.
- We demonstrate the workflow of *AdvCat* on two cybersecurity-critical applications with categorical features: *fake news detection* and *intrusion detection*. Substantial experimental study unveils that the state-of-the-art detection models in both cybersecurity practices are highly vulnerable to the discrete adversarial perturbations. The results raise serious concerns over the trustworthiness of the ML practices in detecting high-stake security events.

It is worth noting that robustifying the vulnerable ML-driven cybersecurity applications is beyond the scope of our study. We therefore don't include robust training and other defense methods against adversarial perturbations in our empirical study. We discuss the related work discussion in Section.2, and introduce *AdvCat* in Section.3. We establish utility analysis of *AdvCat* in Section.4, and conduct empirical study of two applications in Section.5. We conclude the whole paper in Section.6.

II. Related work

Tremendous efforts have been made to *vulnerability measurement of a classifier under evasion attack* [14]–[21].Previous works focus on evaluating classifier robustness against l_p-norm perturbations on continuous data. They all assume adversarial samples locate within a smooth l_p-ball centered at an input instance, which doesn't hold for categorical data. Pioneering works of adversarial attacks on categorical data take advantage of domain-specific knowledge to facilitate the combinatorial search in categorical feature spaces. [8], [12], [22]–[26] focus on replacing words/sentences with manually chosen synonyms, following syntactic guidelines or preserving entailment of texts to mislead text classifiers. [10], [11] narrow down the search range within the image areas containing sensitive contents for image classification. Though the domain-specific constraints effectively improve the search efficiency of adversarial perturbations, the dependency on domain-specific knowledge prevents these evasion attack methods from being extended to new domains. Furthermore, the heuristic search strategies of these methods do not provide provably guarantees of the success of attack.

Most adversarial attack methods on categorical data require white-box access to the target ML models, such as *TextBugger* (the white-box setting) [23], *BERT-Attack* [8], *SememePSO* [24] and *CompAttack* [25]. [7], [9] facilitate adversarial attacks on categorical data by computing gradients of embeddings of categorical features. [27]–[29] adopt edge-flipping and node attribute perturbation to poison graph neural networks (GNN). They appease the NP-hard problem by attacking surrogate models of GNN. These methods are tailored to specific GNN architectures. Though [10]–[13], [23] adapt to the black-box attack setting, they depend heavily on domain knowledge. Notably, [12] generates candidate words using specific word embedding models. [13] requires human annotators to define entailment preserving rules to design adversarial texts. The lack of domain and model-agnostic adversarial perturbation methods becomes the bottleneck of building a generally applicable robustness assessment pipeline.

[7] proposes a black-box and domain-agnostic attack method on categorical inputs, named *FSGS*. It conducts iter-

ative greedy search to select incrementally more categorical features to perturb. However, previous works [7], [9] only established the optimality of *FSGS* when the target classifier has non-negative parameters, which is an unrealistic setting for ML practices. Besides, the computational overheads of *FSGS* increase drastically with higher feature dimensions. Our work answers the open problems. We unveil the rationality of using *FSGS* in *AdvCat* to evaluate the robustness of a general classifier. We further propose two alternative methods in *AdvCat* to provide more economic yet accurate robustness assessment results.

III. AdvCat for Robustness Assessment

A. Workflow of AdvCat

AdvCat provides Robustness Assessment-as-a-Service to ML product owners, e.g., cybersecurity enterprises applying ML-driven intrusion or misinformation detectors. We next introduce the workflow of *AdvCat*, as illustrated in Figure.1.

Privacy-preserving data sharing. An ML service provider shares with *AdvCat* a set of testing instances, which will be used for empirically evaluating the adversarial robustness of the target ML model. The physical sense and concrete values of each categorical feature are completely removed to prevent possible data privacy leak. The categorical values are replaced by integer indexes.

Model-agnostic assessment. Instead of sharing the ML model, the product owner provides the query access, e.g. query APIs to *AdvCat*. It allows *AdvCat* to get the predicted class label and the classification confidence of a submitted testing instance. We leave the label-only scenarios for future study. Besides, we don't assume the availability of any surrogate model to the target ML model as in [25]. Building surrogate models requires extra efforts to collect shadow training data and prior knowledge of the target model architecture, which violates the model-agnostic setting of *AdvCat*.

Domain-agnostic perturbation. The core of *AdvCat* is to construct perturbed instances and test if they can bypass the detector. *AdvCat* constructs adversarial perturbations only based on the queried decision output from the ML model via the API access, without using any domain-specific knowledge. We introduce the domain and model-agnostic assessment methods in Section 3.3. Based on the classification results of perturbed instances, *AdvCat* generates diagnostic reports profiling the adversarial robustness level of the core ML model of the application.

B. Notations and Problem Definition

Let f denote the ML classifier of a cybersecurity-critical application, which predicts a label y_k ($k = 1, ..., K$, $K \geq 2$) for an instance $\boldsymbol{x} = \{x_1, x_2, ..., x_n\}$ of n categorical attributes. Each of the categorical attributes x_i takes any of m ($m \geq 1$) categorical values. In real-world cybersecurity applications, x_i can be one of the behavioural signatures associated with network attacks or keywords of fake news texts. In practices, we cast each categorical value of x_i to a D-dimensional pre-trained embedding vector, e.g., $\boldsymbol{e}^j \in \mathbb{R}^D$, $j = 1, 2, ..., m$. To represent an instance $\boldsymbol{x}$ with the embedding vectors of its category values, we define binary variables $\boldsymbol{b} = \{b_i^j\}$, i=$1, 2, ..., n$, j=$1, 2, ..., m$, where $b_i^j = 1$ when the j-th feature value is present for x_i and $b_i^j = 0$ otherwise. An instance $\boldsymbol{x}$ can then be represented by an $\mathbb{R}^{n*m*D}$ tensor with $x_{\{i,j,:\}} = b_i^j \boldsymbol{e}_i^j$.

We use $\hat{\boldsymbol{b}} = \{\hat{b}_i^j\}$ to indicate the adversarial modifications introduced into $\boldsymbol{b}$. For unperturbed x_i^j, $\hat{b}_i^j = b_i^j$. Otherwise, $\hat{b}_i^j \neq b_i^j$. Depending on the type of attacks to implement (*insertion*, *deletion* or *substitution*), $\hat{b}_i^j$ can have different values. Although f is unknown in our setting, *AdvCat* can query the decision probabilities $f_{y_k}(\boldsymbol{x}, \hat{\boldsymbol{b}})$ of instance $\boldsymbol{x}$ with perturbation $\hat{\boldsymbol{b}}$. For a target ML model f, the problem of evaluating the adversarial robustness on a given input $\boldsymbol{x}$ can be formulated by *set function maximization* in Eq.1.

$$\psi(l) = \max_{\substack{\hat{b}_{i=1...n}, \\ l=\mathit{diff}(b,\hat{b})}} \max_{\hat{b}_i^{j=1...m}} (m_f) \qquad \text{s. t.} \quad m_f \geq 0, \quad |l| \leq \varepsilon \quad (1)$$

where $m_f = \max_{k \in \{1,...,K\}/\{k^*\}} \{f_{y_k}(\boldsymbol{x}, \hat{\boldsymbol{b}})\} - f_{y_{k^*}}(\boldsymbol{x}, \hat{\boldsymbol{b}})$ defines the decision confidence gap over an input instance $\boldsymbol{x}$ between the wrong and correct classification output. k^* is the ground truth label. And $m_f \in [-1, 0)$ and $m_f \geq 0$ correspond to a correct and wrong classification, respectively. Note that we focus on the resilience against *non-targeted evasion attack* in this work. The threat of targeted attack considers additional rank constraints on the classifier's output over different classes, and will be investigated in future.

We aim at seeking a **minimal set of categorical feature perturbations** $l = \mathit{diff}(\boldsymbol{b}, \hat{\boldsymbol{b}})$, with which the **maximum possible difference** between the decision confidence of any wrong class label k and the correct label k^* is larger than 0. The upper limit of the size of the perturbation set l, noted as ε, is used as the budget of adversarial perturbation and a tunable parameter of the assessment service. Higher/lower ε indicates stronger/weaker strength of input perturbation. *AdvCat* adapts ε to customize the perturbation strength in different applications. Given a target ML model, solving Eq.1 to induce $m_f \geq 0$ provokes an alert of flipped decision output on the disturbed input. More/less flipped decisions over the given testing set indicates lower/higher adversarial robustness of the target ML-driven application.

C. Benchmark Algorithms in AdvCat

To solve Eq.1, *AdvCat* proposes to use three optional algorithms (*FSGS*, *SGS* and *UCBS*) to address the NP-hard problem of finding a minimal set of $l = \mathit{diff}(\boldsymbol{b}, \hat{\boldsymbol{b}})$. Given an input $\boldsymbol{x}$, they conduct iterative combinatorial search of feasible perturbations $\hat{\boldsymbol{b}}$ over categorical features, based on the classification output queried from the ML model. They stop the search at iteration t when $m_f(l_t) \geq 0$ (achieving a successful attack with l_t) or the number of modifications $|l_t| > \varepsilon$. To control the attack processing time in practice, we also set a run-time limit (denoted as T_L) to stop the search if no feasible perturbations can be found to make $m_f \geq 0$ for input $\boldsymbol{x}$.

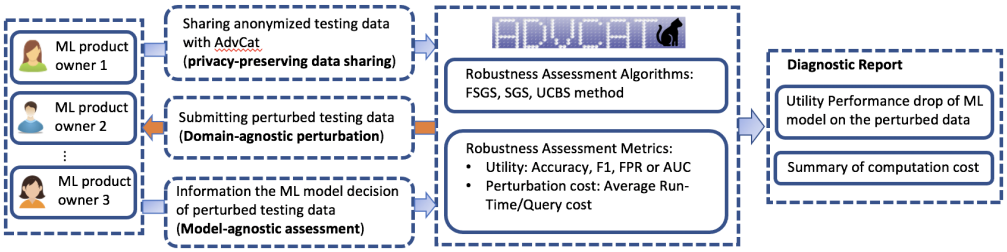

Fig. 1. The workflow of *AdvCat* for Robustness Assessment-as-a-Service

Algorithm 1 FSGS algorithm for robustness assessment

Input: The input $\boldsymbol{x}$, the modifiable features H=$\{a_i\}_{i=1...n}$, the attack budget ε, the run-time limit T_L
Output: The maximum support set S_t pushing $m_f(l_t)$ to violate the tolerance constraint in Eq. 1

1: $S_0 \leftarrow \emptyset$, Run-time $T_R = 0$
2: **for** $t = 0, 1, 2, \ldots$ **do**
3: $T \leftarrow$ *zero-valued vector* $\in \mathbb{R}^n$
4: **for** each $a_i \in H/S_{t-1}$ **do**
5: $T(a_i) = \underset{s \subset S_{t-1}, |s| \leq \varepsilon}{\arg\max} m_f(s \cup a_i)$ (***Inner Level Optimization***)
6: **end for**
7: $\tilde{a} = \underset{a_i}{\arg\max}\ m_f(T(a_i) \cup a_i)$ (***Outer Level Optimization***)
8: $S_t \leftarrow S_{t-1} \cup \{\tilde{a}\}$
9: $l_t = T(\tilde{a}) \cup \tilde{a}$
10: **if** $m_f(l_t) \geq \Gamma$ or Run-time $T_R > T_L$ **then break**
11: **end for**

Algorithm 2 SGS algorithm for robustness assessment

Input: The input $\boldsymbol{x}$, the modifiable features H=$\{a_i\}_{i=1...n}$, the number of features to randomly select r, the attack budget ε, the run-time limit T_L
Output: The maximum support set S_t pushing $m_f(l_t)$ to violate the tolerance constraint in Eq. 1

1: $S_0 \leftarrow \emptyset$, Run-time $T_R = 0$
2: **for** $t = 0, 1, 2, \ldots$ **do**
3: $T \leftarrow$ *zero-valued vector* $\in \mathbb{R}^n$
4: $R \leftarrow$ randomly choose r features from H/S_{t-1}
5: **for** each $a_i \in R$ **do**
6: $T(a_i) = \underset{s \subset S_{t-1}, |s| \leq \varepsilon}{\arg\max} m_f(s \cup a_i)$ (***Inner Level Optimization***)
7: **end for**
8: $\tilde{a} = \underset{a_i}{\arg\max}\ m_f(T(a_i) \cup a_i)$ (***Outer Level Optimization***)
9: $S_t \leftarrow S_{t-1} \cup \{\tilde{a}\}$
10: $l_t = T(\tilde{a}) \cup \tilde{a}$
11: **if** $m_f(l_t) \geq \Gamma$ or Run-time $T_R \geq T_L$ **then break**
12: **end for**

Forward Stepwise Greedy Search (*FSGS*). Given a categorical input $\boldsymbol{x}$, let S denote the support set of perturbed categorical features, which is initially set as an empty set. In each iteration of *FSGS*, we expand S by adding one candidate feature of $\boldsymbol{x}$ that can bring the maximum increase of m_f by perturbing the combination of this feature and the features already in S. More formally, Algorithm 1 gives the pseudo codes of *FSGS*. In each iteration t, *FSGS* first runs the *inner level optimization* (Line 5). For each candidate feature a_i, this is to locate the union of a subset of S_{t-1} and a_i that can induce the maximum increase of m_f. Then in the *outer level optimization*, *FSGS* selects the candidate feature $\tilde{a}$ and adds it to S_t, if the perturbation over $S_t \cup \tilde{a}$ can achieve the largest marginal gain of $m_f(S_t \cup \tilde{a})$ over $m_f(S_{t-1})$ (Line 8).

Stochastic Greedy Search (*SGS*). This is a stochastic variant of the greedy search solution [30]. It has a lower time complexity than *FSGS*. In each iteration of *SGS*, instead of traversing all the candidate features not in S, we randomly choose r of the candidate features. Then we expand the set S by adding one of the r candidate features to reach the largest marginal gain, as *FSGS* does. The details of *SGS* are presented in Algorithm 2.

Upper-Confidence Bound Search (*UCBS*). Different from *FSGS* and *SGS*, this method adopts the criterion of Upper-Confidence Bound (UCB) [31] to explore the combinatorial space of adversarial perturbations. It is a popular Multi-Armed Bandit (MAB) technique, which explicitly balances the exploration and exploitation of the combinatorial search [31]. To employ *UCBS* for robustness assessment, we consider one *categorical feature to perturb* in $\boldsymbol{x}$ as *one arm to pull* in MAB. Unlike *FSGS* and *SGS*, *UCBS* does not need to evaluate the effect of perturbations over all possible combinations of the selected features in each iteration. As given in Algorithm 3, the first step is to obtain the initial reward information after modifying each feature a_i once (line 2-3 in Algorithm 3). The reward $G_{a_i,t}$ is determined by the classifier's prediction on the implemented modification at iteration t, including a_i as well:

$$G_{a_i,t} = \max_{k \in \{1,\ldots,K\}/\{k^*\}} \{f_{y_k}(\boldsymbol{x}, \hat{\boldsymbol{b}}_{a_i,t})\}. \tag{2}$$

The reward $G_{a_i,t}$ has a value in $[0, 1]$. If the feature a_i is not selected in iteration t, the reward $G_{a_i,t}$ will be 0. Over all previous $t-1$ iterations, the experimentally expected reward of perturbing feature a_i is:

$$\hat{\mu}_{a_i,t-1} = \frac{1}{t-1} \sum_{\tau=1}^{t-1} G_{a_i,\tau}. \tag{3}$$

Then $\hat{\mu}_{a_i,t-1}$ is used to build the upper confidence bound function,

$$B_{a_i,T_i,t-1} = \hat{\mu}_{a_i,t-1} + \sqrt{\frac{\alpha * \log t}{2T_{a_i,t-1}}}, \tag{4}$$

α is the parameter to adjust the exploration depth and exploitation level. $T_{a_i,t-1}$ is the accumulated number that the feature a_i been selected during the whole $t-1$ iterations.

Algorithm 3 The UCBS algorithm for robustness assessment

Input: The input $\boldsymbol{x}$, the modifiable features H=$\{a_i\}_{i=1...n}$, the attack budget ε, the run-time limit T_L
Output: the selected features S_t to perturb;
1: $S_0 \leftarrow \emptyset$, $t = 0$, Run-time $T_R = 0$
2: **for** $i_0 \in H$ **do**
3: $G_{a_{i,0}} \leftarrow a_{i,0}$ modify each feature once
4: **end for**
5: **while** $m_f(S_t) \geq 0$ or $|S_t| \leq \varepsilon$ and $T_R \leq T_L$ **do**
6: $\tilde{a} = \arg\max_{a_i \in H} \hat{\mu}_{a_i,t-1} + \sqrt{\frac{\alpha * \log t}{2T_{a_i,t-1}}}$
7: $S_t \leftarrow S_{t-1} \cup \tilde{a}$
8: Update reward $G_{a_i,t}$, $m_f(S_t)$, Run-time T_R
9: t = t + 1
10: **end while**

The second step is the iteration searching process. At each iteration, it will select one feature $\tilde{a}$ with the maximum value of $B_{a_i,T_i,t-1}$ to update S_t (line 6-7 in Algorithm 3). After several iterations of *UCBS*, the combination of the selected features from the modifiable features could cause miss-classification/miss-detection output of the target ML model.

Assessment Algorithm Discussion. Previous studies [7], [9], [11] lack the optimality guarantee of using *FSGS* to evaluate the robustness of a general classifier. It prevents the use of *FSGS* for domain-agnostic robustness evaluation. We address the limit by establishing the provably optimality of *FSGS* for general ML models in Section IV-B.

To appease the concern over the intense computational cost of *FSGS* over high-dimensional categorical feature space, we propose *SGS* and *UCBS* as two computationally economic alternatives for robustness assessment. These two methods are adopted for adversarial robustness assessment for the first time. As reported in Section IV-B, *SGS* and *UCBS* are significantly more efficient than *FSGS* in solving the NP-hard knapsack problem. In addition, the quality analysis in Section IV-B shows that they can also achieve provably accurate assessment results, as in *FSGS*.

As shown in Algorithms 1-3, the three methods can take additional domain-specific rules defining modifiable categorical features as optional inputs. In this way, *AdvCat* can take domain-specific knowledge as plugin to be better customized to concrete applications.

D. Robustness Metrics

Given an ML-driven application, *AdvCat* offers options to use *FSGS*, *SGS* or *UCBS* method alternatively to search for feasible adversarial perturbations over each testing instance. *AdvCat* then measures the security event detection performances before and after injecting the perturbations, in terms of the popular metrics including *Accuracy* (*Acc*), *F1 Score* (*F1*), *AUC score* (*AUC*) or *False Positive Rates* (*FPR*). The smaller/greater decrease of Acc, F1 and AUC after performing perturbations indicates that the ML-driven application is more/less robust against the perturbations. In contrast, a larger *FPR* denotes less resilience to the perturbations and vice versa.

To illustrate the variance of these accuracy metrics caused by the attack, *AdvCat* reports *the percentage decrease / increase of the detection metrics* after the attack is enforced, compared to the adversary-free metrics. They include *Difference of Acc* (*DAcc*↓), *Difference of F1* (*DF1*↓), *Difference of FPR* (*DFPR* ↑), or *Difference of AUC* (*DAUC*↓) respectively. ↓ / ↑ denotes the corresponding drop / rise after perturbation in *AdvCat* compared with the unperturbated performance. For intrusion detection, we consider additionally *Detection Rate* (*DR*) of abnormal system sessions and report *Difference of DR* (*DDR*↓).

AdvCat also reports the expenses of robustness assessment over the testing instances. The expenses are measured by Averaged Query Number and Averaged Running Time (*Runtime*), which are the averaged number of queries that *AdvCat* sends to the ML API and the averaged runtime of perturbing each testing instance. Larger/smaller these meansured expenses represent higher/lower overheads of robustness assessment.

TABLE I
QUERY COMPLEXITY OF *FSGS*, *SGS* AND *UCBS*

Assessment method	Query Complexity
FSGS	$\sum_{t=0}^{T}((n-t)*m*2^t)$
SGS	$\sum_{t=0}^{T}(r*m*2^t)$
UCBS	$n*m+T$

IV. UTILITY ANALYSIS OF *AdvCat*

A. Query Complexity Analysis

In the process of robustness assessment, *AdvCat* iteratively queries the ML model to explore feasible categorical data manipulation. The query cost hence constitutes the majority of the runtime cost of *AdvCat*. The more/less required queries indicate the higher/lower computational complexity of *AdvCat* in practices. For one instance $\boldsymbol{x}$ containing n categorical features and each feature having one of m categorical values, Table I reports the query complexity of *FSGS*, *SGS* and *UCBS* running T iterations. The comparison shows that the greedy based attack methods (*FSGS*, *SGS*) have significantly higher query complexity than *UCBS* method when the running iterations of T and the number of features n become larger.

B. Assessment Quality Analysis

To provide consistently accurate robustness evaluation across different cybersecurity applications and different ML model architectures, it is necessary to analyze the solution quality of *AdvCat* by *FSGS*, *SGS* and *UCBS* in solving the problem in Eq.1. We next establish the analysis that *AdvCat* can provide provably accurate assessment results, regardless of the applications and the choice of ML models .

Definition 1. ***Smoothness of*** f. *Let* $\Omega = (p, q)$, $p, q \in \mathbb{R}^n$ *and the classifier* f: $\mathbb{R}^n \to \mathbb{R}$ *be a Lipschitz-continuous and differentiable function.* f *is* (m_Ω, M_Ω)*-smooth on* Ω*, if for any* $(p, q) \in \Omega$, $m_\Omega \in \mathbb{R}$ *and* $M_\Omega \in \mathbb{R}^+$, $\epsilon = f(q) - f(p) - \langle \nabla f(p), q - p \rangle$ *satisfies:*

$$\frac{m_\Omega}{2}\|q-p\|_2^2 \leq |\epsilon| \leq \frac{M_\Omega}{2}\|q-p\|_2^2. \quad (5)$$

According to [32], most deep neural networks are practically Lipschitiz continuous. These popularly employed classifiers

with finite Lipshitz constants meet the smoothness condition. M_{Ω_ζ} and m_{Ω_ζ} can be computed as local Lipshitz constant and the local strong convexity constant of f respectively. Therefore, the smoothness condition allows much broader and more realistic choices of the classifier, compared to the non-negativity constraint proposed in [7], [9]. Given a classifier satisfying Eq.5, Eq.1 defines in nature a weakly submodular optimization problem. The definition of weak submodularity can be found in [33].

Theorem 1. ***Weak submodularity of the robustness assessment problem.*** *Let* $\Omega_\zeta = \{(\hat{\boldsymbol{b}}, \hat{\boldsymbol{b}'}) : |\mathit{diff}(\boldsymbol{b}, \hat{\boldsymbol{b}})| \leq \zeta, |\mathit{diff}(\boldsymbol{b}, \hat{\boldsymbol{b}'})| \leq \zeta, |\mathit{diff}(\hat{\boldsymbol{b}}, \hat{\boldsymbol{b}'})| \leq \zeta, \zeta \geq 1\}$*, where* $\hat{\boldsymbol{b}}$ *and* $\hat{\boldsymbol{b}'}$ *denote two sets of modifications over* $\hat{\boldsymbol{b}}$*. If the classifier* $f_{y_k}(\boldsymbol{x}, \boldsymbol{b})$ *follows the regularity condition given by* $(m_{k,\Omega_\zeta}, M_{k,\Omega_\zeta})$*-smoothness constraint on* Ω_ζ*, the robustness assessment problem defined in Eq.1 can be formulated respectively as* ***monotone*** γ_ζ***-weakly submodular maximization****.*

Let $\epsilon_k = f_{y_k}(\boldsymbol{x}, \hat{\boldsymbol{b}'}) - f_{y_k}(\boldsymbol{x}, \hat{\boldsymbol{b}}) - \langle \nabla f_{y_k}(\boldsymbol{x}, \hat{\boldsymbol{b}}), \hat{\boldsymbol{b}'} - \hat{\boldsymbol{b}} \rangle$*, and* $\nabla f_{y_k}(\boldsymbol{x}, \boldsymbol{b})_\nu$ *denote the elements of* $\nabla f_{y_k}(\boldsymbol{x}, \boldsymbol{b})$ *corresponding to the difference between* $\hat{\boldsymbol{b}}$ *and* $\hat{\boldsymbol{b}'}$*, where* $\nu = \mathit{diff}(\hat{\boldsymbol{b}}, \hat{\boldsymbol{b}'})$*. The submodularity ratio* γ_ζ *for the robustness assessment problem of Eq.1 is bounded as:*

$$\gamma_\zeta = \min_{k=1,\ldots,K} \{\gamma_{k,\zeta}\} \tag{6}$$

where $\gamma_{k<K,\zeta} \geq \frac{\|\nabla f_{y_k}(\boldsymbol{x},\boldsymbol{b})_\nu\|_2 + m_{k,\Omega_1}|\zeta|/2}{\|\nabla f_{y_k}(\boldsymbol{x},\boldsymbol{b})_\nu\|_2 + M_{k,\Omega_\zeta}|\zeta|/2}$ *for* $\epsilon_k \geq 0$*,*
$\gamma_{k<K,\zeta} \geq \frac{2m_{k,\Omega_\zeta}}{\|\nabla f_{y_k}(\boldsymbol{x},\boldsymbol{b})_\nu\|_2^2}(\|\nabla f_{y_k}(\boldsymbol{x},\boldsymbol{b})_\nu\|_2 - M_{k,\Omega_1}|\zeta|/2)$ *for* $\epsilon_k < 0$*,*
$\gamma_{k^*,\zeta} \geq \frac{2m_{K,\Omega_\zeta}}{\|\nabla f_{y_{k^*}}(\boldsymbol{x},\boldsymbol{b})_\nu\|_2^2}(\|\nabla f_{y_{k^*}}(\boldsymbol{x},\boldsymbol{b})_\nu\|_2 - M_{K,\Omega_1}|\zeta|/2)$ *for* $\epsilon_{k^*} \geq 0$*, and*
$\gamma_{k^*,\zeta} \geq \frac{\|\nabla f_{y_{k^*}}(\boldsymbol{x},\boldsymbol{b})_\nu\|_2 + m_{K,\Omega_1}|\zeta|/2}{\|\nabla f_{y_{k^*}}(\boldsymbol{x},\boldsymbol{b})_\nu\|_2 + M_{K,\Omega_\zeta}|\zeta|/2}$ *for* $\epsilon_{k^*} < 0$*.*

Theorem 2. ***Provably solution quality of FSGS and SGS.*** *For a* $(m_{\Omega_\zeta}, M_{\Omega_\zeta})$*-smooth classifier* f*, the quality of the solution* l *to Eq.1 using FSGS and SGS can be bounded respectively as*

$$\begin{aligned} &|m_f^{\mathit{FSGS}}(l)| \leq e^{-\gamma_\zeta} + (1 - e^{-\gamma_\zeta})|m_f(l^{\mathit{OPT}})| \\ &|m_f^{\mathit{SGS}}(l)| \leq 1 - (1 + 1/\gamma_\zeta)^{-2} + (1 + 1/\gamma_\zeta)^{-2}|m_f(l^{\mathit{OPT}})| \end{aligned} \tag{7}$$

where γ_ζ *is the submodularity ratio in Eq. 6.* l^{OPT} *is the underlying true optimal solutions to the robustness assessment problem of Eq.1.*

Solution quality of FSGS and SGS. The theoretical analysis established in Theorem 1 and 2 shows the rationality of using *FSGS* and *SGS* in *AdvCat* from two perspectives. **First**, if the target model f is $(m_{\Omega_\zeta}, M_{\Omega_\zeta})$*-smooth*, these two methods can provide good approximation quality to the underlying optimal result of the robustness assessment problem, regardless of the architecture of f, as given in Eq.7. **Second**, the solution quality guarantee of *SGS* is less tight compared to that of *FSGS* with the same submodularity ratio. It indicates that the accuracy of the *SGS*-based robustness asseessment may practically present relatively larger variation than *FSGS* across different applications. Nevertheless, in contrast to *FSGS*, *SGS* has a significantly reduced query cost. This makes *SGS* more feasible for scenarios where the query budget to the target ML-based application is limited. These two choices enable *AdvCat* to make a trade-off between the theoretical optimality and the empirical feasibility of the robustness assessment task.

Regret bound of UCBS. Unlike *FSGS* and *SGS*, the solution quality of *UCBS* is not built upon the weak submodularity nature of the robustness assessment problem. We establish the regret bound for *UCBS* to analyse the quality of its solution to the robustness assessment problem in Eq.1. The UCB regret generally indicates the difference between the selected condition and the ideal optimized condition. Concretely in our study, the regret $\triangle_{a_i}$ of *UCBS* is defined as the gap between the adversarial perturbation effects using the selected features and the underlying optimal perturbation effects,

$$\triangle_{a_i} = \mu^* - \mu_{a_i}. \tag{8}$$

where $\mu^* = \max_{1 \leq i \leq n} \mu_{a_i}$ is the largest expected reward. The expected regret $E[R_T]$ bound of *UCBS* under T iterations is [31]:

$$E[R_T] \leq \sum_{\triangle_{a_i} > 0} (\frac{2\alpha}{\triangle_{a_i}} \log T + \frac{\alpha}{\alpha - 2} \triangle_{a_i}). \tag{9}$$

The smaller the regret becomes, the more accurate the *UCBS* robustness assessment result is. The expected regret bound increases slowly as running more iterations T. The *UCBS* can guarantee to search out the high quality feature chain.

V. AdvCat on Security-critical Applications

A. Cybersecurity-critical Applications

Fake news detection. In general, fake news detection models are built on a *news embedding* + *classification* pipeline. Word2Vec [34], Glove [35] and Bert [36] are most widely used for *news embedding*, and CNN, LSTM, and MLP are employed for *classification*. Our study involves the state-of-the-art detection models, including **CNN with Glove** [1], **CNN with Bert**, **MLP with Word2Vec** [37], **MLP with Glove**, **MLP with Bert** [25], [37], [38], **LSTM with Word2Vec**, **LSTM with Glove** [1], [39], and **LSTM with Bert**. CNN and LSTM classifiers take input of words/sentences embedding. MLP takes the embedding for the whole text as input. Few studies consider combinatorial attack of these adversarial words or sentences. In this work, we systematically study the combinatorial attack in fake news detection.

Intrusion detection. Deep Learning-based intrusion detection methods [5], [40] aim to automatically flag suspicious attack events and category different attack techniques using network traffic logs. A vulnerable intrusion detection model to the adversarial perturbations can cause miss-detection of cyber attacks and expose the key infrastructure to the attackers. We involve a state-of-the-art Deep Neural Network-based solution to detecting abnormal events over security event logs, namely *DeepLog* [5] in our benchmark study. **DeepLog** models the sequence of security event logs of normal system sessions using a Long Short-Term Memory (LSTM). Specifically, given the

first M log entries $\{e_{t-M}, ..., e_{t-2}, e_{t-1}\}$ as input, *DeepLog* leverages LSTM to predict the successive log e_t. By treating the integer index of the log-to-predict e_t as a class label, *DeepLog* conducts log prediction as multi-class classification. It adopts the top-K classification scheme: it checks if the target log is one of the top K predictions (the K predicted log with the highest classification confidence). The log-prediction output of *DeepLog* can be extend to detect abnormal sessions of logs containing abnormal behaviours. As reported in [5], *DeepLog* is first trained using benign sessions containing M logs without any security incidents. After that, to detect anomalies, *DeepLog* is deployed over an input M-log long sequence of system logs. Using the first M-1 logs, *DeepLog* then predicts the M-th log and compares the prediction output with the observed log. If they are *inconsistent*, *DeepLog* then tags this sequence as anomaly and vice versa.

B. Assessment Setup

We summarize the datasets used in the assessment pipeline as below. The details about the datasets and assessment settings are provided in Appendix B, due to the space limit. All the experimental implementations are available at https://github.com/codesub/AdvCat.

Fake news detection datasets. We choose two popular datasets: FakeNewsNet [4] and LIAR [1] in the assessment task. **FakeNewsNet** contains 369 fake news and 409 real news crawled from *Politifact* website. We choose 544 samples for training, 77 samples as the validation set and 157 samples as the test set. For sentence paraphrasing in FakeNewsNet, we introduce SCPN [39] to generate neighboring sentences for each sentence in the news text. **LIAR** contains 12,791 short statements collected from *Politifact* website. These samples belong to 6 categories: pants on fire, false, barely true, half-true, mostly true, and true. We regard the first three categories as false and the latter three categories as true. We use 10,240 samples for training and 2,551 samples as the test set. For word paraphrasing in LIAR, we use Paragram-SL999 [41] to generate word paraphrasing neighbors for each word in the texts. We use the maximum fraction of modifiable words/sentences in each news sample as the perturbation budget. For FakeNewsNet, we set the perturbation budget to be 35% and 45% of the length of the news text. For LIAR, we conduct word-level perturbations and set budgets as 10% and 20% of the words in each news samples. We set the attack time limit as 3,600 seconds and 1,000 seconds respectively for FakeNewsNet and LIAR.

Intrusion detection. We also choose two benchmark datasets - HDFS and IPS - for our assessment. **HDFS** [5] contains Hadoop-based executation sessions over 200 Amazon EC2 nodes. Each session contain varied number of MapReduce operation logs. Following the setting in [5], [40], our model was trained on 4,855 normal sessions of logs, tested on 553,366 normal sessions and 15,200 abnormal sessions. **IPS** [6] contains the security event logs collected from 35k sessions by a major security company's intrusion prevention product. Each event represents a network-level (e.g., unauthorized login) or system-level activity (e.g., malware detection) that matches a pre-defined signature. We use the security events from 30k sessions recorded on a single day for training and the events from 5k sessions for testing. In practices, an adversary can change/remove the security logs to mislead ML-based detection [2]. For all our experiments, *AdvCat* hence chooses to modify input logs to *DeepLog* for robustness evaluation. We set the window size of log prediction to 10, the perturbation budget up to 5 event log modifications, and the limit of the run-time of attack to 60 seconds.

C. Robustness Diagnostic Reports

In both fake news detection and intrusion detection applications, we evaluate the adversarial robustness of the security-critical detection models following the strategy of "measurement-by-attack" [42]. We measure the drop of the detection accuracy of the security-critical detection models, when they are exposed to the adversarial attack equipped in *AdvCat*. More specifically, in Table II to Table VI, we show the percentage decrease / increase of the detection accuracy, F1 and False Positive Rate (FPR) compared to those derived in the adversary-free scenario, noted as DAcc, DF1 and DFPR. Besides, we also unveil the average number of required queries and the average running time cost required by each attack method in Fig.2 to Fig.5. These empirical observations help the users of *AdvCat* understand the computational cost of robustness evaluation using the service.

Various state-of-the-art security-critical detection models are significantly vulnerable to adversarial attacks. *AdvCat* provides detailed diagnostic reports on the utility performance of different fakenews detection models in Table II and Table III with the attack budget of 35% and 10% of the length of the sample on FakeNewsNet and LIAR. Table II shows that the different fake news detection methods suffers 40% to 84% decrease of the detection accuracy with respect to the accuracy and F1 scores and **2**-**10** times of the FPR scores obtained at the adversary-free scenario FakeNewsNet. On LIAR, Table III shows that all the involved state-of-the-art detection methods get 57% to 91% decrease of the detection accuracy regarding the accuracy and F1 scores, and up to **2** times of the FPR scores obtained at the adversary-free scenario. The significant decrease of fakenews detection accuracy and increase of FPR scores indicates complete loss of the utility of the state-of-the-art fakenews detection methods facing the attacks.

For the use case of intrusion detection, *AdvCat* first provides the robustness evaluation report of *the log prediction use* of *DeepLog* in Table IV and Table V with the attack budget of 5 (modifying at most 5 logs in each session) on HDFS and IPS for both Top-1 and Top-9 prediction mode. Table IV and Table V show that *DeepLog* suffers from a complete loss of accuracy on HDFS and IPS with respect to both accuracy and F1 score, and significant drop of AUC scores on both datasets.

For detecting abnormal system sessions, *DeepLog* slides a 10-log long window over each session of logs (each session contains varying number of logs, range from 100 to 20k logs). To perform anomaly detection, one session of security event

TABLE II

ROBUSTNESS ASSESSMENT RESULTS ON FAKENEWSNET WITH ATTACK BUDGET =35% AND ATTACK TIME = 3600SEC, REPORTING THE DETECTION PERFORMANCE/PERCENTAGE OF CHANGE AFTER THE ATTACK IS ENFORCED. THE MOST ROBUST MODEL ASSESSED BY EACH ATTACK ALGORITHM IS IN BOLD.

Attack	FEmbed	Model	Acc/DAcc↓	F1/DF1↓	FPR/DFPR↑
FSGS	W2V	MLP	**0.55 (-40%)**	**0.54 (-41%)**	**0.34** (+1030%)
		LSTM	0.43 (-49%)	0.42 (-51%)	0.48 (+242%)
	Bert	MLP	0.13 (-84%)	0.13 (-84%)	0.87 (+335%)
		LSTM	0.24 (-71%)	0.24 (-71%)	0.85 (+347%)
SGS	W2V	MLP	**0.54 (-41%)**	**0.53 (-42%)**	**0.35** (+1060%)
		LSTM	0.44 (-48%)	0.43 (-49%)	0.46 (+228%)
	Bert	MLP	0.13 (-84%)	0.13 (-84%)	0.86 (+330%)
		LSTM	0.25 (-70%)	0.25 (-70%)	0.84 (+342%)
UCBS	W2V	MLP	**0.57 (-38%)**	**0.55 (-40%)**	**0.32** (+700%)
		LSTM	0.45 (-47%)	0.43 (-49%)	0.44 (+667%)
	Bert	MLP	0.19 (-77%)	0.19 (-77%)	0.82 (+310%)
		LSTM	0.26 (-69%)	0.26 (-69%)	0.84 (+342%)

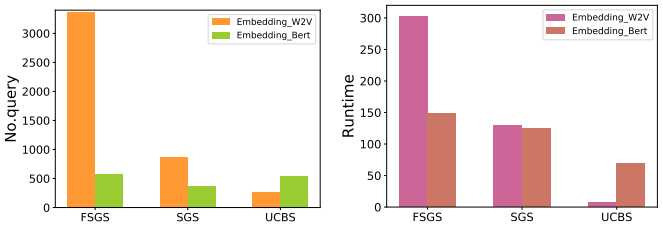

Fig. 2. The assessment expenses on FakeNewsNet dataset. Left: the number of queries. Right: average running time (seconds).

logs can be tagged as normal only if all the sliding windows are detected as normal. Otherwise, the session is raised as an anomaly. *AdvCat* reports *DR* and *FPR* of anomaly detection with the input perturbation, as well as the variation magnitudes of *DR* (*DDR*↓) and *FPR*(*DFPR*↑). In our study, only the HDFS data provide labels for normal and abnormal sessions. We therefore conduct the robustness evaluation of *the anomaly detection use* of *DeepLog* on HDFS only. To demonstrate different attack strength setting, in one session of logs in HDFS dataset, we choose respectively 30% and 60% of the 10-log long sliding windows to inject the adversarial modification of logs. *AdvCat* hence produces the diagnostic report showing the loss of detection rate and rising the FPR scores of session-level anomaly detection after attack on HDFS in Table VI. As observed in the table, we can find anomaly detection using *DeepLog* gets 100% of the detection accuracy loss and 2 times of the FPR scores obtained before being exposed to attacks. This is consistent with the empirical results on the log prediction task, as shown in Table V. The state-of-the-art deep learning-based intrusion detection is prone to small perturbation to the input logs, which can drastically bias the anomaly detection outputs. The results raise a severe concern over the trust of the ML-driven intrusion detection model in practical security-critical applications.

We summarize the perturbation-free log prediction and abnormal session detection accuracy in the appendix. The big gap of the detection performance in fakenews and intrusion detection applications before and after attack shows these security-critical detection models can be arbitrarily misled by

TABLE III

ROBUSTNESS ASSESSMENT RESULTS ON LIAR DATASET OF FAKE NEWS DETECTION WITH ATTACK BUDGET =10% AND ATTACK TIME = 1000SEC. THE MOST ROBUST MODEL ASSESSED BY EACH ATTACK ALGORITHM IS IN BOLD.

Attack	FEmbed	Model	Acc/DAcc↓	F1/DF1↓	FPR/DFPR↑
FSGS	Glove	MLP	0.07 (-88%)	0.06 (-84%)	0.91 (+139%)
		LSTM	0.11 (-81%)	0.15 (-71%)	0.92 (+149%)
		CNN	0.08 (-87%)	0.05 (-91%)	0.87 (+200%)
	Bert	MLP	0.14 (-76%)	0.15 (-76%)	**0.86** (+83%)
		LSTM	0.14 (-77%)	0.18 (-73%)	0.9 (+61%)
		CNN	**0.2 (-67%)**	**0.27 (-60%)**	0.89 (+59%)
SGS	Glove	MLP	0.06 (-90%)	0.04 (-93%)	0.92 (+142%)
		LSTM	0.1 (-82%)	0.14 (-73%)	0.93 (+151%)
		CNN	0.07 (-89%)	0.04 (-93%)	0.89 (+207%)
	Bert	MLP	0.11 (-81%)	0.12 (-81%)	**0.88** (+87%)
		LSTM	0.12 (-74%)	0.17 (-74%)	0.93 (+66%)
		CNN	**0.16 (-74%)**	**0.22 (-68%)**	0.9 (+61%)
UCBS	Glove	MLP	0.12 (-80%)	0.09 (-84%)	0.85 (+124%)
		LSTM	0.17 (-70%)	0.23 (-56%)	0.87 (+135%)
		CNN	0.15 (-76%)	0.07 (-88%)	0.8 (+176%)
	Bert	MLP	0.21 (-64%)	0.27 (-57%)	0.83 (+77%)
		LSTM	0.26 (-57%)	0.32 (-52%)	**0.78 (+39%)**
		CNN	**0.33 (-46%)**	**0.42 (-38%)**	0.79 (+41%)

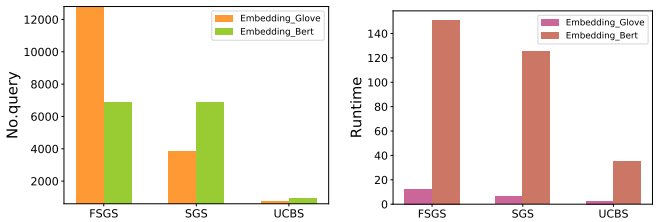

Fig. 3. The assessment expenses on LIAR dataset of fake news detection. Left: the number of queries. Right: average running time (seconds).

small perturbation to the inputs, resulting in complete loss of utility in detecting security incidents. It confirms the necessity of evaluating the adversarial robustness of security-critical ML services before deployment.

The most robust models assessed by each attack algorithm. In Table II and Table III, the most robust fakenews detection model assessed by each attack algorithm is highlighted in bold. On the FakeNewsNet dataset, the three attack algorithms have an agreement that the **MLP with word2vec** is the most robust model. As shown in Table II, this simple model has the smallest drop of accuracy and F1 score, and has the highest accuracy and F1 when attack is enforced. It also has the smallest FPR, although a big increase is caused by the attack. This is an interesting assessment result. Table VII in appendix also shows that **MLP with word2vec** has the best performance on perturbation-free scenario. This simple model can thus been highly recommended for detecting fakenews in application cases that have similar distribution to the FakeNewsNet dataset.

On the other fakenews detection dataset LIAR, all the three attack algorithms have an agreement that the **CNN with Bert** is the most robust model, in items of the accuracy/F1 and accuracy/F1 drop. **MLP with Bert** is reported to have the lowest FPR by FSGS and SGS, while LSTM with Bert is assessed to be the one with the lowest FPR by UCBS. However, as concluded above, all these models had lost utility, as indicated by the super low accuracy and high FPR. **CNN**

TABLE IV
LOG-WINDOW LEVEL ROBUSTNESS ASSESSMENT RESULTS OF *DeepLog* ON HDFS WITH ATTACK BUDGET = 5 AND ATTACK TIME = 60S.

Attack	Top-k	Acc/DAcc↓	F1/DF1↓	AUC/DAUC↓
FSGS	Top-1	0.00 (-100%)	0.00 (-100%)	0.53 (-46%)
	Top-9	0.00 (-100%)	0.00 (-100%)	0.19 (-80%)
SGS	Top-1	0.00 (-100%)	0.00 (-100%)	0.59 (-40%)
	Top-9	0.085 (-91%)	0.15 (-85%)	0.19 (-80%)
UCBS	Top-1	0.10 (-89%)	0.18 (-81%)	0.57 (-42%)
	Top-9	0.36 (-64%%))	0.53 (-46%)	0.27 (-73%)

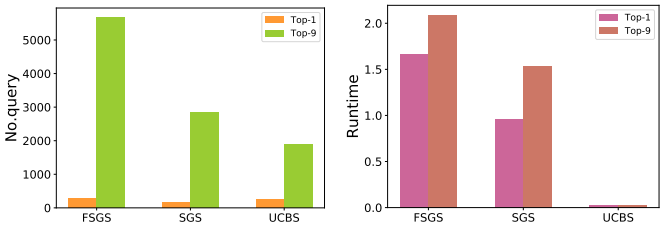

Fig. 4. The assessment expenses on HDFS dataset of intrusion detection. Left: the number of queries. Right: average running time (seconds).

TABLE V
LOG-WINDOW LEVEL ROBUSTNESS ASSESSMENT RESULTS OF *DeepLog* ON IPS WITH ATTACK BUDGET = 5 AND ATTACK TIME = 60S.

Attack	Top-k	Acc/DAcc↓	F1/DF1↓	AUC/DAUC↓
FSGS	Top-1	0.00 (-100%)	0.00 (-100%)	0.48 (-44%)
	Top-9	0.0024 (-100%)	0.0049 (-100%)	0.47 (-47%)
SGS	Top-1	0.00 (-100%)	0.00 (-100%)	0.59 (-40%)
	Top-9	0.0050 (-100%)	0.0099 (-100%)	0.50 (-43%)
UCBS	Top-1	0.00 (-100%)	0.00 (-100%)	0.58 (-32%)
	Top-9	0.0051 (-100%)	0.010 (-99%)	0.42 (-52%)

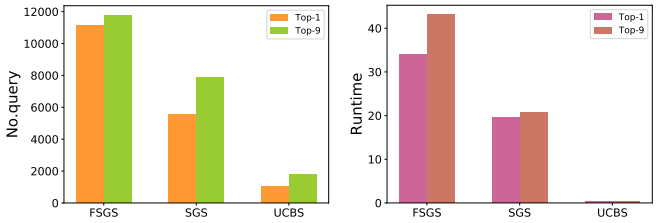

Fig. 5. The assessment expenses on IPS dataset of intrusion detection. Left: the number of queries. Right: average running time (seconds).

with Bert is perhaps recommended as the best candidate model if robustness improvement strategies can be applied.

The intrusion detection models have been assessed to have 100% loss of detection accuracy, as shown in Table IV, V and VI. Their robustness should be all improved.

Computation cost measurement. On FakeNewsNet, we give the attack cost on the average number of queries and average running time with word2vec/Bert embedding and MLP/LSTM classifiers in Fig. 2. On LIAR, we show the attack cost in the same measures regarding Glove/Bert embedding, equipped with MLP, LSTM and CNN-based classifiers in Fig. 3. It is interesting to find that attacking the Bert embedding-based models cost less queries than the word2vec or Glove embedding based detection models. However, the running-time cost of attacking Bert is higher than the other two embedding techniques. For one thing, as reported in [43], Bert-based embedding is highly prone to the synonym-based adversarial attack methods in text classification, which can be the potential reason behind the observed low query cost. For the other thing, modifying one original word during attack requires to compute the Bert embedding of the synonym substituting the original word. Comparing to word2vec and Glove, computing the Bert embedding requires more time, which explains the relatively higher running time of attacking Bert-based models.

For the intrusion detection use cases, on HDFS and IPS, we demonstrate the attack cost on the average of the average number of query and average running time in Top-1 and Top-9 based models from Fig. 4 and Fig. 5. Success of the Top-9 prediction only requires the true log-to-predict exists in the Top-9 ranked outputs according to the prediction confidence. In contrast, the Top-1 prediction requires the true log-to-predict has the highest prediction confidence. Therefore, Top-1 is less stable facing adversarial perturbations. As unveiled, we find that attacking the Top-1 mode prediction models on both HDFS and IPS needs less average time and query number than the Top-9 mode. As observed in Fig. 2 to Fig. 5, we show a trade-off between attack effectivenes and computational efficiency existing in the three attack methods offered by *AdvCat*. Among all the three attack methods, on one hand, UCBS and FSGS spend consistently the least and most average running time and query cost respectively. SGS ranks in between them. UCBS shows the order of magnitudes higher efficiency than FSGS in perturbing the target detection model. On the other hand, across different detection models and security-critical use cases, FSGS can achieve more stronger attack effects, causing significantly lower detection accuracy and F1 score and higher FPR values than SGS and UCBS. The users of *AdvCat* can choose the proper attack method upon their budgets of computational resources for practical robustness evaluation tasks. In general, the experimental results show that *AdvCat* can provide the robustness alert to potential vulnerable detection models before being deployed in practices.

VI. CONCLUSION

AdvCat is established as a domain- and model-agnostic pipeline to evaluate the adversarial robustness of ML-driven cybersecurity-critical applications with categorical features. We address the challenges from two perspectives. First, we propose to integrate three domain-agnostic assessment methods into *AdvCat*. Based on them, *AdvCat* can provide assessment-as-a-service to any ML-driven cybersecurity applications, without accessing the target model and privacy-sensitive contents of the application data. Second, we build the quality and computational cost analysis of these methods, which guarantee the use of the assessment pipeline for general classifiers. Via substantial empirical study, we demonstrate the use of *AdvCat* over fake news and intrusion detection use cases. The results unveil the state-of-the-art ML-driven cybersecurity applications are highly vulnerable to adversarial attacks. In future, we plan to extend the use of *AdvCat* to more real-world cybersecurity use cases.

TABLE VI
SESSION-LEVEL ROBUSTNESS ASSESSMENT OF *DeepLog* ON HDFS DATASET WITH VARIED PERTURBED LOG-WINDOWS.

Method	Attack	Window	Top-1		Top-9	
			FPR/DFPR↑	DR/DDR↓	FPR/DFPR↑	DR/DDR↓
Deeplog	FSGS	60% window	1.00 (+212%)	0.00 (-100%)	1.00 (+99%)	0.00 (-100%)
		30% window	1.00 (+212%)	0.00 (-100%)	1.00 (+99%)	0.00 (-100%)
	SGS	60% window	1.00 (+212%)	0.00 (-100%)	0.99 (+100%)	0.00 (-100%)
		30% window	1.00 (+212%)	0.00 (-100%)	0.99 (+100%)	0.00 (-100%)
	UCBS	60% window	0.99 (+219%)	0.0006 (-100%)	0.97 (+96%)	0.0028 (-100%)
		30% window	0.99 (+219%)	0.0007 (-100%)	0.94 (+93%)	0.026 (-95%)

ACKNOWLEDGEMENTS

The research reported in this paper was partially supported by funding from King Abdullah University of Science and Technology (KAUST).

REFERENCES

[1] W. Y. Wang, "" liar, liar pants on fire": A new benchmark dataset for fake news detection," *arXiv preprint arXiv:1705.00648*, 2017.
[2] F. Pierazzi, F. Pendlebury, J. Cortellazzi, and L. Cavallaro, "Intriguing properties of adversarial ml attacks in the problem space," in *SP*, 2020, pp. 1332–1349.
[3] I. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," in *ICLR*, 2015.
[4] K. Shu, D. Mahudeswaran, S. Wang, D. Lee, and H. Liu, "Fakenewsnet: A data repository with news content, social context, and spatiotemporal information for studying fake news on social media," *Big data*, vol. 8, no. 3, 2020.
[5] M. Du, F. Li, G. Zheng, and V. Srikumar, "Deeplog: Anomaly detection and diagnosis from system logs through deep learning," in *ACM CCS*, 2017.
[6] Y.Shen and G.Stringhini, "ATTACK2VEC: Leveraging temporal word embeddings to understand the evolution of cyberattacks," in *USENIX Security*, 2019.
[7] Y. Wang, Y. Han, H. Bao, Y. Shen, F. Ma, J. Li, and X. Zhang, "Attackability characterization of adversarial evasion attack on discrete data," in *ACM KDD*, 2020.
[8] L. Li, R. Ma, Q. Guo, X. Xue, and X. Qiu, "Bert-attack: Adversarial attack against BERT using BERT," in *EMNLP*, 2020.
[9] L. Qi, L. Wu, P. Chen, A. Dimakis, I. Dhillon, and M. Witbrock, "Discrete attacks and submodular optimization with applications to text classification," in *SysML*, 2019.
[10] S. Narodytska, N.and Kasiviswanathan, "Simple black-box adversarial attacks on deep neural networks," in *CVPR Workshops*, 2017.
[11] F. Croce and M. Hein, "Sparse and imperceivable adversarial attacks," in *ICCV*, 2019.
[12] D. Jin, Z. Jin, J. Zhou, and P. Szolovits, "Is BERT really robust? A strong baseline for natural language attack on text classification and entailment," in *AAAI*, 2020.
[13] J. T. and A. V., "Adversarial attacks against fact extraction and verification," 2019.
[14] A. Fawzi, S. Moosavi-Dezfooli, and P. Frossard, "Robustness of classifiers: From adversarial to random noise," in *NIPS*, 2016.
[15] M. Hein and M. Andriushchenko, "Formal guarantees on the robustness of a classifier against adversarial manipulation," in *NeurIPS*, 2017.
[16] J. Cohen, E. Rosenfeld, and Z. Kolter, "Certified adversarial robustness via randomized smoothing," in *ICML*, 2019.
[17] T. Weng, H. Zhang, H. Chen, Z. Song, C. Hsieh, D. Boning, I. S.Dhillion, and L. Daniel, "Towards fast computation of certified robustness for relu networks," in *ICML*, 2019.
[18] A. Sinha, H. Namkoong, and J. Duchi, "Certifiable distributional robustness with principled adversarial training," in *ICLR*, 2018.
[19] Z. Shi, H. Zhang, K. Chang, M. Huang, and C. Hsieh, "Robustness verification for transformers," *arXiv preprint arXiv:2002.06622*, 2020.
[20] D. Yin, K. Ramchandran, and P. Bartlett, "Rademacher complexity for adversarially robust generalization," in *ICML*, 2019.
[21] Z. Tu, J. Zhang, and D. Tao, "Theoretical analysis of adversarial learning: A minimax approach," in *NeurIPS*, 2019.
[22] P. Yang, J. Chen, C. Hsieh, J. Wang, and M. Jordan, "Greedy attack and gumbel attack: Generating adversarial examples for discrete data." *J. Mach. Learn. Res.*, vol. 21, no. 43, 2020.
[23] J. Li, S. Ji, T. Du, B. Li, and T. Wang, "Textbugger: Generating adversarial text against real-world applications," in *NDSS*, 2019.
[24] Y. Zang, F. Qi, C. Yang, Z. Liu, M. Zhang, Q. Liu, and M. Sun, "Word-level textual adversarial attacking as combinatorial optimization," in *ACL*, 2020.
[25] B. Wang, C. Xu, S. Wang, Z. Gan, Y. Cheng, J. Gao, A. H. Awadallah, and B. Li, "Adversarial glue: A multi-task benchmark for robustness evaluation of language models," *arXiv preprint arXiv:2111.02840*, 2021.
[26] B. Wang, H. Pei, B. Pan, Q. Chen, S. Wang, and B. Li, "T3: Tree-autoencoder constrained adversarial text generation for targeted attack," in *EMNLP*, 2020.
[27] A. Bojchevski and S. Günnemann, "Adversarial attacks on node embeddings via graph poisoning," in *ICML*, 2019.
[28] A. Bojchevski and S. Günnemann, "Certifiable robustness to graph perturbations," in *NeurIPS*, 2019.
[29] A. Akbarnejad and S. Günnemann, "Adversarial attacks on node embedding via graph poisoning," in *ICML*, 2019.
[30] L. Chen, M. Feldman, and A. Karbasi, "Weakly submodular maximization beyond cardinality constraints: Does randomization help greedy?" in *ICML*, 2017.
[31] P. Auer, N. Cesa-Bianchi, and P. Fischer, "Finite-time analysis of the multiarmed bandit problem," *Machine learning*, vol. 47, no. 2, 2002.
[32] M. Jordan and A. G. Dimakis, "Exactly computing the local lipschitz constant of relu networks," 2021.
[33] E. R. Elenberg, R. Khanna, S. Dimakis, A.and Negahban *et al.*, "Restricted strong convexity implies weak submodularity," *The Annals of Statistics*, vol. 46, no. 6B, 2018.
[34] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *arXiv preprint arXiv:1301.3781*, 2013.
[35] J. Pennington, R. Socher, and C. D. Manning, "Glove: Global vectors for word representation," in *EMNLP*, 2014.
[36] J. Devlin, M. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.
[37] Y. Dou, K. Shu, C. Xia, P. S. Yu, and L. Sun, "User preference-aware fake news detection," *arXiv preprint arXiv:2104.12259*, 2021.
[38] R. Maheshwary, S. Maheshwary, and V. Pudi, "Generating natural language attacks in a hard label black box setting," in *AAAI*, 2021.
[39] M. Iyyer, J. Wieting, K. G., and L. Z., "Adversarial example generation with syntactically controlled paraphrase networks," *arXiv preprint arXiv:1804.06059*, 2018.
[40] T. van Ede, H. Aghakhani, N. Spahn, R. Bortolameotti, A. Cova, M.and Continella, M. van Steen, A. Peter, C. Kruegel, and G. Vigna, "DeepCASE: Semi-Supervised Contextual Analysis of Security Events," in *IEEE S&P*, 2022.
[41] J. Wieting, M. Bansal, and K. Gimpel, K.and Livescu, "From paraphrase database to compositional paraphrase model and back," *Transactions of the Association for Computational Linguistics*, vol. 3, 2015.
[42] B. Wang, C. Xu, S. Wang, S. Wang, Z. Gan, Y. Cheng, J. Gao, A. Awadallah, and B. Li, "Adversarial glue: A multi-task benchmark for robustness evaluation of language models," in *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*, J. Vanschoren and S. Yeung, Eds., vol. 1, 2021. [Online]. Available: https://datasets-benchmarks-proceedings.neurips.cc/paper/2021/file/335f5352088d7d9bf74191e006d8e24c-Paper-round2.pdf
[43] D. Jin, Z. Jin, J. T. Zhou, and P. Szolovits, "Is bert really robust? a strong baseline for natural language attack on text classification

and entailment," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 05, pp. 8018–8025, Apr. 2020.

[44] F. Pendlebury, F. Pierazzi, R. Jordaney, J. Kinder, and L. Cavallaro, "TESSERACT: Eliminating experimental bias in malware classification across space and time," in *USENIX Security*, 2019.

[45] K. Grosse, N. Papernot, P. Manoharan, M. Backes, and P. McDaniel, "Adversarial examples for malware detection," in *ESORICS*, 2017, pp. 62–79.

APPENDIX

We show the fake news detection performances including *Accuracy*, *F1 Score*, *AUC Score* or *False Positive Rate*. We evaluated these performance before and after attack and use *DAcc*, *DF1*, *DAUC* or *DFPR* respectively to denote the descending value or ascending value after attack. In Table.VII, we provide the perturbation-free performances of different fake news detection models over the two datasets. They are obtained without injecting adversarial perturbations, which measures the adversary-free utility baseline of fake news detection.

We summarize the adversary-free performances of *DeepLog* as in Table.VIII. We measure *Acc*, *F1*, *FPR* or *AUC* of the log-prediction results. For session-level anomaly detection on the HDFS data, *DR* and *FPR* of the abnormal session detection result using *DeepLog* are 0.63 and 0.31 for the top-1 prediction mode, and 0.55 and 0.0013 for the top-9 prediction mode.

TABLE VII
PERTURBATION-FREE FAKE NEWS DETECTION ACCURACY

Dataset	FEm	Model	ACC	Pre	Rec	F1	AUC	FPR
FakeNewsNet	W2V	MLP	0.92	0.95	0.86	0.92	0.96	0.04
		LSTM	0.85	0.84	0.85	0.85	0.90	0.14
	Bert	MLP	0.82	0.78	0.84	0.82	0.88	0.20
		LSTM	0.83	0.81	0.85	0.82	0.88	0.19
LIAR	Glove	MLP	0.59	0.55	0.55	0.55	0.50	0.38
		LSTM	0.57	0.54	0.50	0.52	0.47	0.37
		CNN	0.62	0.60	0.52	0.56	0.46	0.29
	Bert	MLP	0.59	0.62	0.65	0.63	0.53	0.47
		LSTM	0.60	0.60	0.74	0.66	0.55	0.56
		CNN	0.61	0.61	0.76	0.68	0.56	0.56

We conduct the experimental study using PyTorch on a Ubuntu workstation with two NVIDIA 3090 GPUs.

FakeNewsNet contains news evaluated by exports or journalists and tagged as fake or real news. FakeNewsNet also contains the information about social contexts and spatiotemporal information. [37] adopts Graph Neural Networks integrating these additional records with news contents and historical posts of the users spreading the news to help classification. In our study, we only focus on the adversarial perturbation over the news contents.

LIAR is also collected from *Politifact* website. Different from FakeNewsNet, LIAR contains mostly short statements without meta attributes. It collects 12,791 samples and they are classified into 6 categories: pants on fire, false, barely true, half-true, mostly true, and true. For our task, we regard the first three categories as false and the latter three categories as true. We use 10,240 samples for training and 2551 samples as the test set.

As the news samples of FakeNewsNet contain longer texts than those of LIAR, we apply sentence-level perturbations for FakeNewsNet and word-level perturbations for LIAR. For word paraphrasing in LIAR, we use Paragram-SL999 [41] to generate word paraphrasing neighbors for each word in the texts. For sentence paraphrasing in FakeNewsNet, we introduce SCPN [39] to generate neighboring sentences for each sentence in the news text. Note that we treat the neighboring words/sentences as alternative categorical values of each word/sentence in the original news texts. They define the domain-specific rules to bound the perturbation search space over the news texts. While conducting assessment, *AdvCat* only needs to know the indexes of these neighboring words/sentences. In other words, the testing news samples are anonymized.

For FakeNewsNet and LIAR, we set the time limit as 3,600 seconds and 1,000 seconds respectively. As different samples have different numbers of words or sentences, we use the maximum fraction of modifiable words/sentences in each news sample as the perturbation budget. For each dataset, we test with two budget settings to assess the robustness under different degrees of attack. For FakeNewsNet, we conduct sentence-level perturbations. There are over 30% of the news samples containing less than 3 sentences. Therefore, for each testing news sample in FakeNewsNet, we choose the perturbation budget to be 35% and 45% of the length of the news text. For LIAR, we conduct word-level perturbations and set budgets as 10% and 20% of the words in each news samples.

Intrusion detection. In our study, *AdvCat* conducts the robustness assessment task of *Deeplog* on the HDFS and IPS dataset, following the setting in [5], [6], [40]. **HDFS** was collected by executing Hadoop-based map-reduce operations on over 200 Amazon EC2 nodes, and was labeled by intrusion detection domain experts. About 2.9 % of the 11.2M system log entries gathered are anomalous [5]. Our model was trained on 4,855 normal sessions, tested on 553,366 normal sessions and 15,200 abnormal sessions. **IPS** [6] contains security

TABLE VIII
PERTURBATION-FREE LOG-WINDOW LEVEL DETECTION PERFORMANCE OF INTRUSION DETECTION

Dataset	Top-k	Model	ACC	Pre	Rec	F1	AUC
HDFS	Top-1	DeepLog	0.91	1.00	0.91	0.96	0.99
	Top-9	DeepLog	0.99	1.00	1.00	1.00	0.99
IPS	Top-1	DeepLog	0.58	1.00	0.59	0.74	0.87
	Top-9	DeepLog	0.95	1.00	0.96	0.98	0.87

TABLE IX
ROBUSTNESS ASSESSMENT RESULTS ON FAKENEWSNET WITH ATTACK BUDGET =45% AND ATTACK TIME = 3600S

Attack	FEmbed	Model	Acc/DAcc↓	F1/DF1↓	FPR/DFPR↑
FSGS	W2V	MLP	0.46 (-50%)	0.43 (-53%)	0.40 (+900%)
		LSTM	0.35 (-59%)	0.32 (-62%)	0.48 (+243%)
	Bert	MLP	0.10 (-89%)	0.10 (-89%)	0.88 (+340%)
		LSTM	0.17 (-80%)	0.17 (-79%)	0.86 (+353%)
SGS	W2V	MLP	0.47 (-48%)	0.44 (-52%)	0.38 (+850%)
		LSTM	0.36 (-58%)	0.33 (-62%)	0.47 (+236%)
	Bert	MLP	0.12 (-85%)	0.12 (-85%)	0.84 (+320%)
		LSTM	0.17 (-80%)	0.17 (-79%)	0.86 (+353%)
UCBS	W2V	MLP	0.45 (-57%)	0.42 (-53%)	0.40 (+900%)
		LSTM	0.36 (-58%)	0.33 (-61%)	0.49 (+250%)
	Bert	MLP	0.14 (-83%)	0.14 (-83%)	0.85 (+325%)
		LSTM	0.18 (-78%)	0.17 (-79%)	0.87 (+358%)

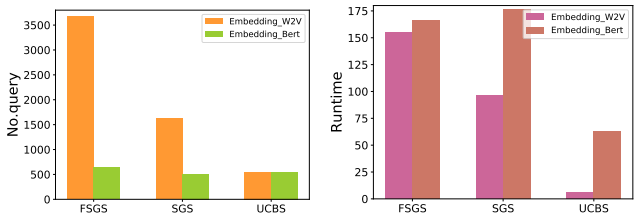

Fig. 6. The assessment expenses on FakeNewsNet dataset of fakenews detection . Left: the number of queries. Right: average running time.

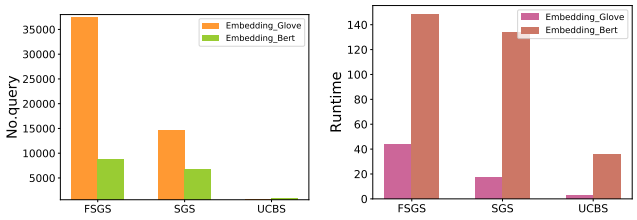

Fig. 7. The assessment expenses on LIAR dataset of fakenews detection . Left: the number of queries. Right: average running time.

TABLE X
ROBUSTNESS ASSESSMENT RESULTS ON LIAR WITH ATTACK BUDGET = 20% AND ATTACK TIME = 1000S.

Attack	FEmbed	Model	Acc/DAcc↓	F1/DF1↓	FPR/DFPR↑
FSGS	Glove	MLP	0.05 (-92%)	0.04 (-93%)	0.94 (+147%)
		LSTM	0.1 (-82%)	0.13 (-75%)	0.93 (+151%)
		CNN	0.06 (-90%)	0.03 (-95%)	0.97 (+234%)
	Bert	MLP	0.11 (-81%)	0.13 (-79%)	0.91 (+94%)
		LSTM	0.12 (-80%)	0.16 (-76%)	0.92 (+64%)
		CNN	0.18 (-70%)	0.24 (-65%)	0.9 (+61%)
SGS	Glove	MLP	0.05 (-91%)	0.04 (-93%)	0.95 (+150%)
		LSTM	0.09 (-84%)	0.13 (-75%)	0.94 (+154%)
		CNN	0.06 (-90%)	0.03 (-95%)	0.91 (+213%)
	Bert	MLP	0.1 (-83%)	0.11 (-83%)	0.9 (+0.91%)
		LSTM	0.11 (-82%)	0.15 (-77%)	0.92 (+64%)
		CNN	0.16 (-74%)	0.23 (-66%)	0.91 (+63%)
UCBS	Glove	MLP	0.09 (-85%)	0.07 (-87%)	0.93 (+145%)
		LSTM	0.15 (-74%)	0.19 (-63%)	0.87 (+135%)
		CNN	0.12 (-81%)	0.06 (-89%)	0.85 (+193%)
	Bert	MLP	0.18 (-69%)	0.21 (-67%)	0.82 (+74%)
		LSTM	0.18 (-70%)	0.23 (-65%)	0.84 (+50%)
		CNN	0.27 (-56%)	0.36 (-47%)	0.82 (+46%)

events of a major security company's intrusion prevention product. Meta-information associated with a security event on an endpoint is recorded when this product detects network-level (e.g., unauthorized login) or system-level activity (e.g., malware detection) that matches a pre-defined signature. We use the anonymized machine ID to reconstruct a series of security events detected in a given machine and discard it after the reconstruction process is done. We use the security events from 30k sessions recorded on a single day for training and 5k sessions of events for testing. We strictly require that training and test data to come from different machines so as not to introduce evaluation bias [44]. Note that we measure both the top-1 and top-9 of the prediction as normal. For all our experiments, we set the attack budget to 5, window size to 10, and the time limit to 60 seconds. More specifically, *DeepLog* is performed over a sequence (a window) containing 10 logs. They use the 10 logs in this window and predict the successive log.

Modifying logs for robustness assessment. *AdvCat* applies modifications over the input logs to *DeepLog*-based log prediction models for robustness assessment. Pioneering research efforts [2], [45] in the security community unveil empirically modifying and/or removing code segments and network traffic packets can camouflage malicious incidents and evade ML-driven security event detection. In practices, an adversary has many hacking techniques, e.g. man-in-the-middle attacks and advanced persistent attacks to compromise the infrastructures transferring and/or storing these system event logs. He can then change/remove the logs to mislead the ML-based detection engine [2].