FraudBench: Protocol-Sensitive Benchmarking of Adversarial Robustness for Financial Risk Assessment

Xitong Zeng, Zhaoge Bi, Yitian Yang, and Huaming Chen
School of Electrical and Computer Engineering
The University of Sydney
Sydney, Australia

Quan Z. Sheng
School of Computing
Macquarie University
Sydney, Australia

***Abstract*—Machine learning models are widely used in financial fraud and credit-risk detection, yet their adversarial robustness remains difficult to evaluate because financial tabular data involve domain-specific constraints, severe class imbalance, and asymmetric attacker capability. We argue that, in this setting, robustness is not only an attribute of the model, but also an attribute of the evaluation protocol. Different ways of enforcing constraints and capability can lead to substantially different robustness conclusions. This paper presents FraudBench, a protocol-sensitive benchmark for adversarial robustness evaluation in financial fraud and credit-risk detection. Rather than treating domain constraints as post-hoc validity checks, FraudBench evaluates the same dataset–model–attack–defence setting under three matched protocols: unconstrained attacks, post-hoc feasibility filtering, and deployment-aware constraint-integrated attacks. FraudBench covers four public financial datasets, and evaluates neural, tree-based, and ensemble models using three attack settings. Our results show that robustness conclusions are highly protocol-sensitive. On Lending Club Loan Data under the white-box setting, post-hoc filtering leaves only 3.7 feasible-flipped examples on average, whereas in-attack projection with attacker mutability masking produces 2,832.3 feasible-flipped examples under the same perturbation budget. The results on IEEE-CIS further show that feasibility and attacker capability are separate axes, while black-box evaluation shows that protocol choice can alter model-family rankings. These findings suggest that fraud robustness evaluation should report predictive degradation and attack feasibility jointly, and should incorporate domain constraints into attack generation rather than treating them as post-processing checks. The code repository is available at https://github.com/iHaydenzZ/FraudBench.**

***Index Terms*—adversarial robustness, financial fraud detection, tabular data, benchmark**

I. Introduction

Financial fraud is a sustained and growing problem. Global card payment fraud losses reached $33.41 billion in 2024, with card-not-present (CNP) transactions continuing to create substantial operational risk for issuers and merchants [1], [2]. Since fraudulent transactions are rare relative to legitimate activity, manually reviewing every transaction is economically infeasible. As a result, the screening of incoming transactions is overwhelmingly delegated to machine learning models that operate on financial tabular features.

Adversarial robustness in this setting differs from the image-classification setting that originally motivated much of adversarial machine learning [3]–[5]. In image tasks, perturbations are commonly formalised as bounded changes within an L_p ball. In tabular financial fraud detection, however, each feature has an operational meaning. A transaction amount must remain positive, categorical attributes must preserve valid category structure, and loan-related variables may be linked by arithmetic or contractual relationships. Moreover, attacker capability is asymmetric: an adversary may influence transaction-facing or self-reported fields, but cannot freely modify credit-bureau records, platform-internal risk scores, or historical account attributes. These properties make fraud robustness a domain-specific problem rather than a direct application of generic adversarial evaluation.

Another complication arises from the statistical properties of fraud data. Fraudulent cases are rare, and the positive-class rate can be far below one percent in credit-card transaction datasets [6]. Under such imbalance, accuracy is often misleading: a classifier may obtain high accuracy by predicting most records as legitimate while failing to identify fraudulent transactions. Financial fraud detection also relies heavily on tabular models such as gradient-boosted decision trees, including XGBoost and LightGBM, which remain strong baselines for structured data [7]–[9]. A robustness benchmark for fraud detection should therefore not be restricted to neural models or accuracy-based evaluation. It should include production-relevant tree-based models, and prioritize metrics such as PR-AUC that reflect rare-class detection quality.

A separate challenge is the validity of adversarial examples. In ordinary adversarial classification, a successful attack is often defined only by whether the perturbed input changes the model prediction. In financial fraud detection, this criterion is not sufficient. A perturbed record that violates domain constraints is not a realistic adversarial example. Rather, it becomes an invalid financial record that can be rejected before model inference. Prior work on constrained tabular attacks has shown that tabular robustness requires constraints over feature ranges, categorical validity, feature mutability, and cross-feature relationships [10]–[13]. Fraud-specific robust-

ness evaluation must measure both predictive degradation and whether the generated adversarial records remain feasible.

One way to study such robustness is through standardised benchmarks. RobustBench [14] and ARES [15] have demonstrated the value of fixed protocols and public comparability in image-domain adversarial robustness. In the tabular setting, TabularBench [16] provides an important benchmark with constrained attack infrastructure and a large-scale leaderboard for tabular deep learning models. Fraud detection has also been studied through public datasets and fraud-focused benchmarks covering credit-card transactions, online retail transactions, loan default, and broader abuse detection settings [17]–[19]. In parallel, several studies have examined adversarial evasion against fraud detectors and attacker strategies in banking and payment systems [20]–[23].

However, existing benchmarks still do not fully answer a central operational question for financial fraud detection: *can an attacker, constrained by realistic feature mutability and domain validity requirements, evade the detector?* General adversarial-robustness benchmarks are mostly designed around image-domain threat models, while existing fraud benchmarks mainly focus on natural predictive performance or privacy-aware evaluation rather than adversarial robustness under domain constraints [19], [24]. Tabular robustness benchmarks provide valuable constraint-aware infrastructure, but their evaluation is not organised around the fraud-specific dimensions of severe class imbalance, production-relevant tree models, asymmetric attacker capability, and feasibility-aware attack success. In particular, post-hoc constraint filtering can measure how many generated attacks are infeasible, but it cannot determine how successful an attacker would be if domain constraints were enforced during attack generation. Thus, robustness conclusions may reflect the evaluation protocol rather than the intrinsic robustness of the model.

To address this gap, we present FraudBench, a protocol-sensitive adversarial robustness benchmark for tabular financial fraud and credit-risk detection. The key idea is to treat the evaluation protocol itself as a first-class experimental variable. FraudBench is designed to isolate how robustness conclusions change when the same experimental setting is evaluated under three protocols: unconstrained attack generation, post-hoc feasibility filtering, and deployment-aware constraint-integrated attack generation. The benchmark therefore treats protocol sensitivity as the main object of study, rather than reporting robustness as a single attack score in which domain constraints are ignored. This paper makes four contributions:

- We introduce FraudBench, a reproducible benchmark for protocol-sensitive adversarial robustness evaluation in financial risk assessment, covering four public datasets, production-relevant model families, white-box and black-box attacks, and standard defense configurations.
- We formulate a deployment-aware fraud robustness evaluation protocol that distinguishes unconstrained attack success, post-hoc filterability, and constraint-aware attacker success through in-attack projection and attacker-capability masks.
- We show that protocol choice can materially change robustness conclusions. On LCLD, the feasible-flipped count changes from 3.7 under post-hoc filtering to 2,832.3 under projection plus mutability masking. On IEEE-CIS, projection increases feasible attacks but masking suppresses them, demonstrating that feasibility and capability are separate axes.
- We analyse model-family and defense conclusions under the same protocol view. Square Attack shows that protocol choice can change model selection across MLP, XGBoost, and ensemble models, while deployment-aware defense results show that adversarial training is the strongest evaluated defence and simple z-score input validation is not a reliable defence.

II. Related Work

A. Adversarial Robustness Benchmarks

Adversarial robustness has been extensively benchmarked in computer vision, where attacks are commonly formulated as bounded perturbations under an L_p norm. Classical attacks such as FGSM and PGD established the standard evaluation setting, while later attack suites such as AutoAttack improved reliability by combining multiple strong attacks [3]–[5]. RobustBench and ARES further showed the value of fixed protocols, strong attacks, and public comparability for robustness evaluation [14], [15]. However, these benchmarks are mainly designed for image-domain threat models. Their assumptions do not directly transfer to financial fraud detection, where input features have operational meanings and invalid perturbations can often be rejected before model inference.

B. Constrained Tabular Adversarial Robustness

Tabular adversarial robustness requires constraints that are absent or less explicit in image classification. Prior work has formalised tabular constraints in terms of feature immutability, domain validity, relational dependencies, and consistency between features [10]. In financial data, such constraints are central: transaction amounts must remain positive, one-hot encoded categorical variables must remain valid, and loan-related fields may satisfy arithmetic relationships. Perturbations that violate these constraints are not realistic adversarial examples, because they correspond to invalid financial records.

Several attacks have been proposed for constrained tabular spaces. FENCE [11] combines gradient-based updates with constraint repair, while MOEVA [12] searches for feasible adversarial examples through multi-objective evolutionary optimisation. CAPGD and CAA further extend projected-gradient-style attacks to constrained tabular data [13]. TabularBench [16] is the closest general-purpose benchmark, providing constraint specifications, constrained attack pipelines, and a large-scale leaderboard for tabular deep learning models. Nevertheless, its evaluation is not organised around fraud-specific deployment requirements such as class imbalance, tree-based production models, asymmetric attacker capability, and feasibility-aware attack success. FraudBench builds on this

line of work by measuring how robustness conclusions change across evaluation protocols.

C. Financial Fraud Detection Models and Benchmarks

Machine learning has long been used in financial fraud detection, including credit-card fraud, online transaction fraud, loan default prediction, and synthetic transaction fraud settings [6], [17]–[19]. A key challenge is *severe class imbalance*: fraudulent cases are rare, and in some datasets the positive-class rate is far below one percent [6]. Under such imbalance, accuracy can be misleading because a classifier may obtain high accuracy by predicting almost all records as legitimate. PR-AUC is therefore more informative for fraud detection.

Another important property is the continued strength of tree-based models. Gradient-boosted decision trees such as XGBoost and LightGBM remain strong baselines for structured tabular data and are widely used in practical risk-modelling pipelines [7]–[9]. Deep tabular models have received substantial attention, but they do not consistently dominate tree-based methods on typical tabular datasets. A fraud-robustness benchmark that evaluates only differentiable neural networks therefore misses an important part of the deployed model landscape. Fraud-focused datasets and benchmarks have also supported progress in this area. Public resources such as CCFD, IEEE-CIS, Sparkov, and Lending Club data provide useful evaluation settings for different fraud and risk-modelling tasks [6], [17], [18]. The Fraud Dataset Benchmark unifies multiple fraud and abuse datasets under common loaders and evaluation procedures, while private fraud benchmarking has been studied in graph settings under differential privacy constraints [19], [24]. These resources standardise clean fraud detection evaluation, but do not systematically evaluate adversarial robustness under domain constraints, attack–defence configurations, and fraud-specific robustness metrics.

D. Deployment Gaps in Fraud Robustness Evaluation

Prior work on adversarial fraud detection has examined evasion attacks on banking models, reinforcement-learning-based attacker strategies, imbalanced tabular attacks, and transferable attacks against credit-card fraud detectors [20]–[23]. These studies show that fraud models can be vulnerable to adaptive manipulation, but the literature remains fragmented across datasets, model families, attacks, defences, and metrics.

This paper addresses four deployment gaps. First, fraud robustness should be evaluated with rare-class-sensitive metrics rather than accuracy alone. Second, production-relevant tree models should be evaluated alongside neural and ensemble models. Third, post-hoc filtering of invalid adversarial examples is not equivalent to generating feasible adversarial examples during the attack. Fourth, standard defences such as adversarial training, input validation, and ensembling should be compared under the same fraud-specific protocol. FraudBench is designed to measure these gaps directly by evaluating the same experimental setting under unconstrained attacks, post-hoc feasibility filtering, and deployment-aware constraint-integrated attacks.

TABLE I
FRAUDBENCH EXPERIMENTAL COVERAGE

Axis	Coverage
Datasets	CCFD, IEEE-CIS, LCLD, Sparkov
Models	MLP, XGBoost, heterogeneous ensemble
Attacks	CAPGD, Square Attack, HopSkipJump partial cross-check
Defences	None, adversarial training, input validation, ensemble
Seeds	42, 123, 456
Primary metric	PR-AUC
Feasibility metrics	Aggregate feasibility, feasible-flipped count, filtered success rate

III. FRAUDBENCH DESIGN

A. Benchmark Scope

FraudBench is a config-driven benchmark for evaluating adversarial robustness in tabular financial fraud and credit-risk detection. While no particular fraud detector is introduced, FraudBench grounds the robustness conclusions with the evaluation protocol. It follows the domain relevant pattern with a unified experiment design, specifying a dataset, model family, attack, defence configuration, perturbation budget, and random seed. The unified pipeline loads the dataset, applies a stratified split, preprocesses the features, trains the model, evaluates clean performance, constructs the constraint schema, generates adversarial examples, checks feasibility, and logs predictive and feasibility-aware metrics. Figure 1 provides an overview of the FraudBench pipeline.

Table I summarises the experimental axes covered by FraudBench. The benchmark combines four public financial datasets, three model families, white-box and black-box attacks, four defence configurations, and three random seeds. This design allows the same setting to be evaluated under different robustness protocols rather than comparing unrelated model–dataset–attack combinations.

B. Datasets and Constraint Structure

Table II summarises the four public datasets used in FraudBench, covering credit-card fraud, online transaction fraud, loan default, and simulated transaction fraud. LCLD is a credit-risk/default task rather than a transaction-fraud dataset. It is included because it provides rich financial-domain constraints and a realistic setting in which borrower-controlled and institution-controlled fields differ. The selected datasets differ strongly in positive-class prevalence, feature semantics, and constraint structure, allowing the benchmark to test whether robustness conclusions hold across both highly imbalanced fraud tasks and a less imbalanced credit-risk task.

The datasets are treated as static tabular binary classification tasks rather than temporal forecasting tasks. FraudBench first draws a stratified 10% subsample for computational tractability and then applies a stratified 60/20/20 train/validation/test split. The cached split is reused across model, attack, and defence configurations within each seed. Numerical variables are standardised with `StandardScaler`, categorical vari-

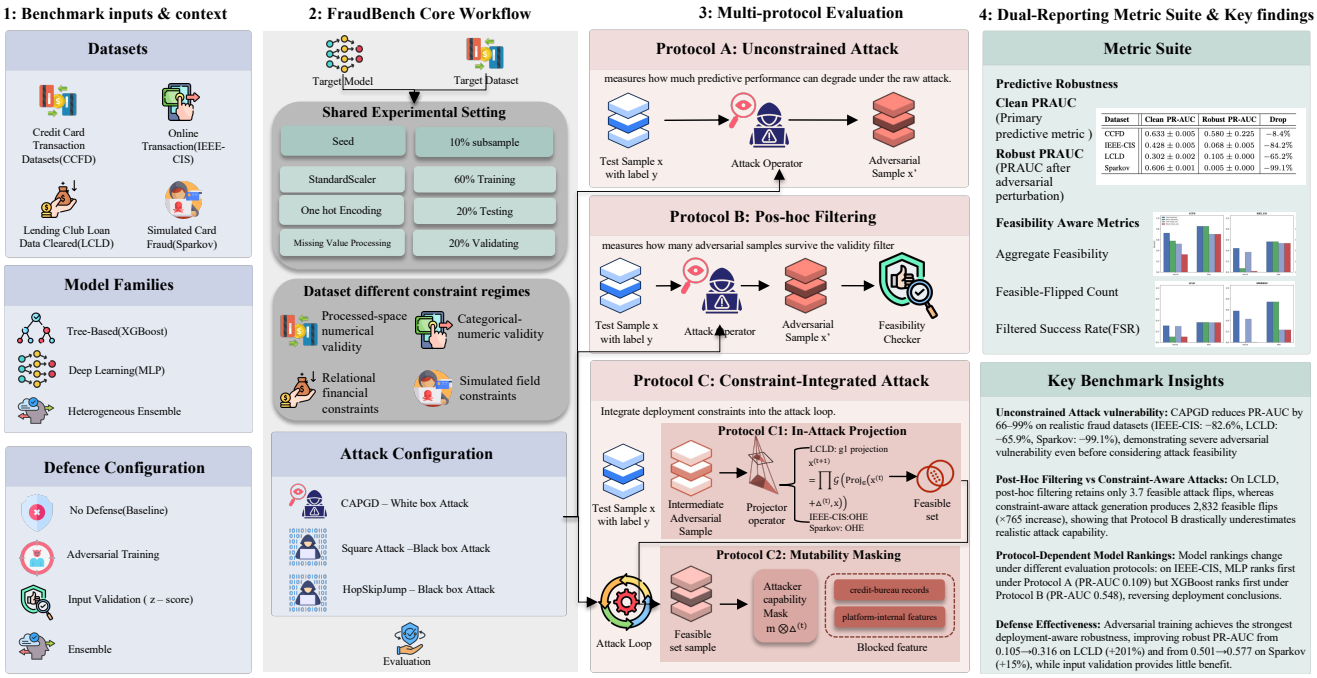

Fig. 1. Overview diagram of the FraudBench benchmark.

TABLE II
FRAUDBENCH DATASETS

Dataset	Samples	Raw	Proc.	Pos.	Scenario
CCFD	284,807	30	30	0.17%	Card fraud
IEEE-CIS	590,540	392	537	3.5%	Online fraud
LCLD	1,340,968	63	188	20.0%	Loan default
Sparkov	1,296,675	11	75	0.6%	Simulated card fraud

ables are one-hot encoded, and missing values are imputed according to feature type.

The datasets provide different constraint regimes. CCFD is largely PCA-anonymised and therefore has no meaningful semantic constraint schema beyond processed-space numerical validity. It is used as a negative-control setting in which post-hoc filtering should not change protocol conclusions. IEEE-CIS contains one-hot validity constraints for categorical transaction fields such as `ProductCD`, `card4`, and `card6`, as well as non-negativity or positivity constraints for amount and count-like features. LCLD contains the richest explicit relational structure, including the loan instalment amortisation formula, inequalities such as `open_acc` $\leq$ `total_acc`, and one-hot validity for loan term. Sparkov contains simulated transaction fields with one-hot constraints for state, category, and gender, together with range constraints for transaction amount, city population, and merchant location.

C. Models, Attacks, and Defences

FraudBench evaluates three model families. The tree model XGBoost [7] is included because gradient-boosted trees remain a common production choice for tabular fraud detection. The neural model is a two-hidden-layer multilayer perceptron with ReLU activations, trained using class-weighted binary cross-entropy and the Adam optimiser. The heterogeneous ensemble model combines logistic regression, XGBoost, and the MLP through soft voting by averaging predicted fraud probabilities. The heterogeneous ensemble is treated as a model-family configuration and as a cross-model robustness comparison, rather than as a neural-model defence.

The primary white-box attack is Constrained Adaptive Projected Gradient Descent (CAPGD), following constrained tabular robustness work [13]. In FraudBench, unmodified CAPGD is used as the Protocol A and Protocol B attack generator for differentiable neural evaluation. Protocol C variants extend the same attack loop with dataset-specific projection or masking operators. Unless otherwise specified, CAPGD uses an L_∞ perturbation budget of $\epsilon = 0.1$ in processed feature space and ten attack iterations.

FraudBench also includes Square Attack [25] and HopSkipJump [26] as black-box companions. Square Attack is the cross-model black-box backbone because it is run over the four datasets and three model families with multi-seed coverage. HopSkipJump is included as a decision-based cross-check, but its coverage is partial due to computational cost. Since XGBoost is non-differentiable, gradient-based CAPGD

TABLE III
PROTOCOL COVERAGE MATRIX

Dataset	Main Constraints	C Operator	CAPGD	Square	HSJ
CCFD	None/PCA	None	A/B	A/B	Partial
IEEE-CIS	OHE, non-neg.	OHE+mask	A/B/C1/C2	A/B	Partial
LCLD	g_1, inequalities, OHE	g_1+mask	A/B/C1/C2	A/B	Partial
Sparkov	OHE, ranges	OHE+mask	A/B/C1/C2	A/B	Partial

TABLE IV
COMPARISON OF EVALUATION PROTOCOLS

Protocol	Constraints in attack	Post-hoc filtering	Interpretation
A	×	×	Unconstrained adversarial robustness
B	×	✓	Feasible subset after filtering unconstrained attacks
C	✓	✓	Constraint-aware attacker success under deployment conditions

results on the tree model are treated as clean-reference or no-op rows rather than valid white-box robustness estimates. Tree-model robustness is therefore read primarily from black-box attacks.

FraudBench evaluates four defence configurations: no defence, adversarial training, input validation, and heterogeneous ensembling. The no-defence setting serves as the baseline. Adversarial training augments neural-model training with CAPGD-generated adversarial examples and uses the same perturbation scale as evaluation [27]. Input validation is implemented as z-score clipping based on training-set statistics: numerical values outside a three-standard-deviation range are clipped before prediction. The ensemble defence replaces the single classifier with the soft-voting ensemble described above.

D. Protocol Coverage

Table III summarises the coverage of the protocol mechanisms. The table makes the benchmark scope explicit. Protocol C is implemented for the main constraint mechanisms needed by LCLD, IEEE-CIS, and Sparkov, while CCFD serves as a negative-control dataset because its PCA-anonymised features do not support a semantic constraint catalogue.

IV. DEPLOYMENT-AWARE EVALUATION PROTOCOL

FraudBench evaluates adversarial robustness through three matched protocols. The protocols share the same dataset split, preprocessing pipeline, model, attack budget, and random seed. They differ only in how domain constraints and attacker capability are handled during evaluation. This design isolates the effect of the evaluation protocol itself.

A. Protocol A: Unconstrained Attack

Protocol A is the unconstrained adversarial baseline. Given a clean test sample x with label y, the attack operator generates an adversarial sample x' within the perturbation budget. No domain constraint is enforced during attack generation, and no infeasible sample is removed after the attack. Protocol A therefore measures how much predictive performance can degrade under the raw attack.

This protocol is useful as a stress test, but it is not a complete fraud-robustness estimate. In tabular financial data, unconstrained perturbations can easily produce invalid records. For example, a continuous attack may turn a one-hot categorical block into fractional values or break the consistency between a loan amount, interest rate, term, and instalment value. Such samples may still fool the model, but they are not necessarily valid financial records.

B. Protocol B: Post-Hoc Filtering

Protocol B uses the same adversarial samples generated by Protocol A, but applies a feasibility checker after attack generation. Let $\mathcal{G}(x')$ denote a constraint checker that returns one if x' satisfies all implemented domain constraints and zero otherwise. Protocol B keeps only the feasible subset:

$$\mathcal{S}_B = \{x' : \mathcal{G}(x') = 1\}. \tag{1}$$

Predictive metrics under Protocol B are then computed on the feasible adversarial subset.

Protocol B answers a specific question: among the adversarial examples produced by an unconstrained attack, how many survive the validity filter? It does not answer whether a constraint-aware attacker could generate feasible adversarial examples directly. Therefore, a low feasible-flipped count under Protocol B may reflect attack-generation inefficiency rather than genuine model robustness.

C. Protocol C: Constraint-Aware Attack Generation

Protocol C integrates deployment constraints into the attack loop. Instead of generating arbitrary adversarial samples and filtering them afterward, Protocol C repairs or restricts the intermediate adversarial sample after each attack step. In FraudBench, Protocol C is implemented through two mechanisms: in-attack projection and mutability masking. These mechanisms correspond to two different deployment requirements. Projection enforces feasibility, while masking enforces attacker capability.

1) C1: In-Attack Projection: C1 applies projection after each attack step. The attack first proposes an intermediate adversarial sample, and the projection operator maps selected feature groups back into the feasible set:

$$x^{(t+1)} = \Pi_{\mathcal{G}}\left(\mathrm{Proj}_{\epsilon}\left(x^{(t)} + \Delta^{(t)}, x\right)\right), \tag{2}$$

where $\Delta^{(t)}$ is the attack update, Proj_{ϵ} enforces the perturbation budget, and $\Pi_{\mathcal{G}}$ repairs the implemented domain constraints.

On LCLD, the main projection is the g_1 projection, which enforces the instalment amortisation constraint:

$$\mathrm{installment} = \frac{\mathrm{loan\_amnt} \cdot r(1+r)^t}{(1+r)^t - 1}, \tag{3}$$

where r is the monthly interest rate and t is the loan term in months. After each attack step, the perturbed `installment` value is overwritten by the value implied by the current

`loan_amnt`, `int_rate`, and `term`. The attacker may still perturb loan-related variables, but the resulting record must remain consistent with the loan-payment formula.

On IEEE-CIS and Sparkov, the main projection restores one-hot encoding validity. For a categorical block $z = (z_1, \ldots, z_K)$, the projection is

$$\Pi_{\mathrm{OHE}}(z)_j = \begin{cases} 1, & j = \arg\max_k z_k, \\ 0, & \text{otherwise.} \end{cases} \tag{4}$$

This prevents continuous attacks from producing invalid categorical mixtures, such as a card type that is partly Visa and partly Mastercard.

2) C2: In-Attack Projection with Mutability Masking: C2 extends C1 by adding an attacker-capability mask. A mutability mask $m \in \{0,1\}^d$ specifies which processed features the attacker is allowed to modify. The attack update is masked before the sample is updated:

$$\Delta^{(t)} \leftarrow m \odot \Delta^{(t)}. \tag{5}$$

Immutable features therefore remain unchanged throughout the attack. The mask is derived from a raw-feature threat model and then mapped into processed feature space after preprocessing. If a raw categorical feature is mutable, all one-hot dimensions derived from it are mutable. If the raw feature is immutable, the corresponding processed dimensions are frozen. This captures the asymmetric attacker capability in fraud settings. A borrower may influence self-reported fields such as loan amount, income, purpose, or term, but cannot directly edit credit-bureau records or platform-internal features. Similarly, an online transaction attacker may control transaction-facing fields but not opaque internal risk features.

D. Protocol Execution

Fig. 2 summarises the execution of the three protocols. Protocol A returns the unconstrained adversarial sample. Protocol B uses the same sample but evaluates only the feasible subset. Protocol C1 inserts projection into the attack loop, and Protocol C2 further applies mutability masking before each projected update.

E. Metrics

FraudBench reports both predictive and feasibility-aware metrics. Predictive metrics include clean PR-AUC and robust PR-AUC. PR-AUC is the primary predictive metric because fraud datasets are often severely imbalanced, making accuracy unreliable as a ranking measure. Robust PR-AUC is computed on adversarially perturbed inputs.

Feasibility-aware metrics include aggregate feasibility, feasible-flipped counts, and filtered success rate. A flipped example is an input whose prediction changes after attack. A feasible-flipped example is both prediction-flipped and valid under all domain constraints. The filtered success rate is defined as

$$\mathrm{FSR} = \frac{\#\{\text{flipped} \cap \text{feasible}\}}{\#\{\text{flipped}\}}. \tag{6}$$

Input: Test sample x, label y, trained model f, attack operator $\mathcal{A}$, constraint checker $\mathcal{G}$, projection operator $\Pi_{\mathcal{G}}$, mutability mask m, protocol $p \in \{A, B, C1, C2\}$
Output: Adversarial sample x', prediction flip indicator Flip, feasibility indicator Feas

Initialise $x^{(0)} \leftarrow x$
for $t = 0$ to $T-1$ **do**
  Generate attack update $\Delta^{(t)} \leftarrow \mathcal{A}(f, x^{(t)}, y)$
  if $p = C2$ **then**
    Apply mutability mask: $\Delta^{(t)} \leftarrow m \odot \Delta^{(t)}$
  end if
  Update sample: $\tilde{x}^{(t+1)} \leftarrow x^{(t)} + \Delta^{(t)}$
  Project to budget: $\tilde{x}^{(t+1)} \leftarrow \mathrm{Proj}_{\epsilon}(\tilde{x}^{(t+1)}, x)$
  if $p = C1$ or $p = C2$ **then**
    Apply in-attack projection: $x^{(t+1)} \leftarrow \Pi_{\mathcal{G}}(\tilde{x}^{(t+1)})$
  else
    Set $x^{(t+1)} \leftarrow \tilde{x}^{(t+1)}$
  end if
end for
Set $x' \leftarrow x^{(T)}$
Compute prediction flip: Flip $\leftarrow \mathbb{I}\{f(x') \neq f(x)\}$
Compute feasibility: Feas $\leftarrow \mathcal{G}(x')$
if $p = B$ **then**
  Keep x' only if Feas $= 1$. Otherwise discard it from the feasible subset
end if
return x', Flip, Feas

Fig. 2. FraudBench evaluation protocols. Protocol A runs an unconstrained attack. Protocol B applies post-hoc filtering to unconstrained attacks. Protocol C1 applies in-attack projection after each attack step. Protocol C2 combines in-attack projection with attacker mutability masking.

This metric is not the positive-class ratio. It is the proportion of successful prediction flips that also satisfy all domain constraints.

FraudBench reports feasibility metrics alongside PR-AUC because the two capture different aspects of robustness. PR-AUC measures ranking degradation under attack, whereas feasible-flipped count measures how many successful attacks remain plausible under the dataset's business rules and attacker-capability assumptions. A model can show severe robust-PR-AUC degradation even when most attacks are infeasible, or it can appear safe under post-hoc filtering while remaining vulnerable to an attacker that generates feasible adversarial records directly.

V. Experimental Setup

All experiments use the cached stratified subsample and split described above. Unless otherwise specified, the main CAPGD setting uses L_∞ perturbations with $\epsilon = 0.1$ in processed feature space and ten attack steps. This budget should be interpreted as a first-order robustness stress test in standardised feature space rather than as a complete economic attack-cost model. Future work should replace or complement this threat model with monetary, profit-based, and attacker-cost-aware perturbation budgets.

Experiments are replicated over seeds 42, 123, and 456. Results are logged to CSV registries with dataset, model, defence, attack, protocol, seed, perturbation budget, clean predictive metrics, adversarial predictive metrics, feasibility metrics, failed constraint labels, and runtime. The table-level artifact used for this manuscript consists of the CAPGD grid registry and the Square Attack model-family registry, which

TABLE V
CLEAN AND CAPGD-ROBUST PR-AUC ON THE NEURAL MODEL WITHOUT DEFENCE

Dataset	Clean PR-AUC	Robust PR-AUC	Drop
CCFD	0.683 ± 0.189	0.598 ± 0.262	-12.5%
IEEE-CIS	0.433 ± 0.033	0.075 ± 0.013	-82.6%
LCLD	0.308 ± 0.003	0.105 ± 0.000	-65.9%
Sparkov	0.626 ± 0.033	0.005 ± 0.000	-99.1%

are sufficient to verify the numerical aggregates reported in the tables.

VI. EXPERIMENTAL RESULTS

A. Unconstrained Attack Vulnerability

Table V reports the headline neural-model results without defence. CAPGD causes large degradation on IEEE-CIS, LCLD, and Sparkov, while CCFD is less affected but exhibits high seed variance due to its extremely small positive class.

These results support the use of PR-AUC as the primary predictive metric. Since fraud positives are rare, accuracy can remain high even when the model fails to rank fraudulent records above legitimate ones. However, PR-AUC alone is insufficient for robustness evaluation because it measures ranking degradation but not whether the adversarial records remain feasible.

B. Post-Hoc Filtering versus Constraint-Aware Attack Generation

Table VI compares the FraudBench protocols under CAPGD at $\epsilon = 0.1$ with no defence. Protocol B PR-AUC is not directly comparable to Protocol A PR-AUC as an intrinsic robustness score because infeasible adversarial samples are removed before evaluation. Its role is to quantify the filterability of unconstrained attacks. C1 denotes in-attack projection, and C2 denotes in-attack projection with mutability masking. The visualisation is shown in Figure 3.

On CCFD, Protocol A and Protocol B are identical because no semantic constraint catalogue is defined. This makes CCFD a negative-control case: when there are no domain constraints, post-hoc filtering does not change the result.

On LCLD, Protocol A reaches robust PR-AUC 0.105 ± 0.000, but only 3.7 flipped examples remain feasible. Protocol B reports a higher robust PR-AUC because infeasible adversarial records are removed before evaluation. This should be interpreted as attack filterability, not intrinsic model robustness. In contrast, C1 raises the feasible-flipped count to 2,071.0 by integrating the instalment formula during attack generation, and C2 further raises it to 2,832.3 with full feasibility. Thus, post-hoc filtering retains only a tiny fraction of the attacks that a constraint-aware attacker can generate directly.

IEEE-CIS and Sparkov show the same qualitative issue through categorical validity constraints. On IEEE-CIS, one-hot projection in C1 raises the feasible-flipped count from 0.0 to 93.7, while adding a conservative mutability mask in C2 reduces the count to 4.3. On Sparkov, Protocol A produces near-zero robust PR-AUC but zero feasible-flipped attacks, whereas C1 and C2 produce feasible-flipped attacks directly. These results show that one-hot validity is not a minor post-processing detail. Instead, it changes the operational meaning of attack success.

TABLE VI
CAPGD PROTOCOL COMPARISON AT $\epsilon = 0.1$ WITH NO DEFENCE

Dataset	Prot.	Robust PR-AUC	Agg. Feas.	Feas.-Flip.	FSR
CCFD	A	0.598 ± 0.262	1.000	0.0	–
CCFD	B	0.598 ± 0.262	1.000	0.0	–
IEEE-CIS	A	0.075 ± 0.013	0.000	0.0	0.000
IEEE-CIS	B	0.433 ± 0.033	1.000	0.0	–
IEEE-CIS	C1	0.079 ± 0.005	0.453	93.7	0.474
IEEE-CIS	C2	0.412 ± 0.033	1.000	4.3	1.000
LCLD	A	0.105 ± 0.000	0.001	3.7	0.001
LCLD	B	0.306 ± 0.003	0.998	3.7	1.000
LCLD	C1	0.105 ± 0.000	0.776	2,071.0	0.732
LCLD	C2	0.105 ± 0.000	1.000	2,832.3	1.000
Sparkov	A	0.005 ± 0.000	0.000	0.0	0.000
Sparkov	B	0.626 ± 0.033	1.000	0.0	–
Sparkov	C1	0.440 ± 0.040	1.000	17.0	1.000
Sparkov	C2	0.501 ± 0.033	1.000	13.0	1.000

C. Protocol-Dependent Model-Family Rankings

Table VII reports Square Attack results across datasets and model families. These results are also visualised by Figure 4. Since Square Attack does not require gradient access, it is the main cross-model black-box attack and is especially important for evaluating XGBoost and the heterogeneous ensemble.

TABLE VII
SQUARE ATTACK RESULTS ACROSS DATASETS AND MODEL FAMILIES. A DENOTES UNCONSTRAINED ATTACK, AND B DENOTES POST-HOC FILTERING.

Dataset	Model	A PR-AUC	B PR-AUC	A Feas.	A Flip.	Time
CCFD	MLP	0.651 ± 0.216	0.651 ± 0.216	1.000	0.0	8.1
CCFD	XGBoost	0.774 ± 0.223	0.774 ± 0.223	1.000	0.7	32.1
CCFD	Ensemble	0.728 ± 0.237	0.728 ± 0.237	1.000	0.7	53.2
IEEE-CIS	MLP	0.109 ± 0.024	0.432 ± 0.034	0.218	192.0	29.5
IEEE-CIS	XGBoost	0.037 ± 0.024	0.548 ± 0.033	0.032	109.3	623.4
IEEE-CIS	Ensemble	0.063 ± 0.016	0.516 ± 0.025	0.125	213.7	783.2
LCLD	MLP	0.121 ± 0.003	0.305 ± 0.002	0.389	2,797.3	29.2
LCLD	XGBoost	0.179 ± 0.022	0.360 ± 0.002	0.243	290.3	540.3
LCLD	Ensemble	0.139 ± 0.005	0.367 ± 0.003	0.380	2,836.3	477.7
Sparkov	MLP	0.003 ± 0.000	0.625 ± 0.033	0.440	163.0	52.7
Sparkov	XGBoost	0.109 ± 0.009	0.678 ± 0.063	0.195	102.7	369.8
Sparkov	Ensemble	0.074 ± 0.010	0.507 ± 0.079	0.014	76.3	526.8

The key information in Table VII is not only the robust PR-AUC value, but also the model-family conclusion induced by the protocol. Table IX summarises the best model under Protocols A and B. On IEEE-CIS, the most robust model under Protocol A is MLP, whereas Protocol B ranks XGBoost first.

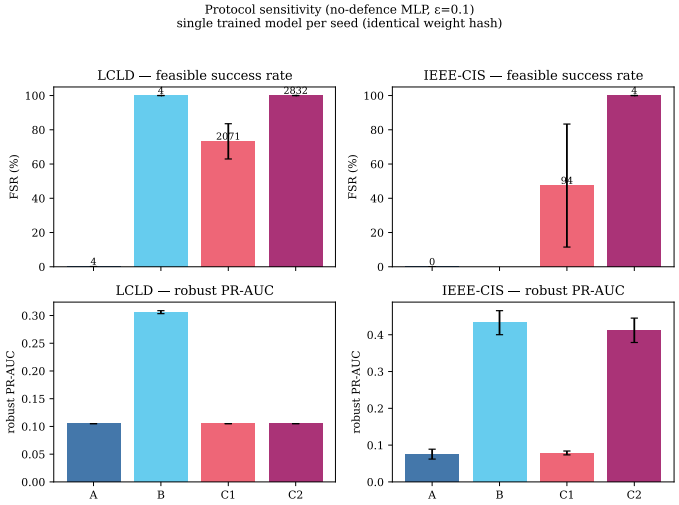

Fig. 3. Protocol sensitivity under CAPGD at $\epsilon = 0.1$ on the no-defence MLP. The top panels show filtered success rate (FSR), with the annotated numbers indicating feasible-flipped counts. The bottom panels show robust PR-AUC. On LCLD, Protocol B retains only a few feasible-flipped attacks from the unconstrained attack, while Protocols C1 and C2 generate feasible attacks directly. On IEEE-CIS, C1 increases feasible attack success through one-hot projection, whereas C2 shows the suppressive effect of mutability masking.

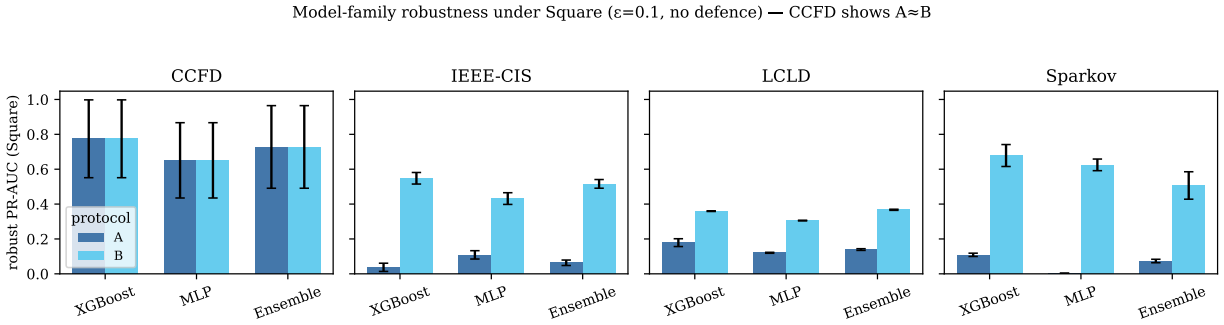

Fig. 4. Model-family robustness under Square Attack at $\epsilon = 0.1$ with no defence. Bars show robust PR-AUC under Protocol A (unconstrained attack) and Protocol B (post-hoc filtering), with error bars indicating one standard deviation across seeds. CCFD shows almost identical Protocol-A and Protocol-B results because no semantic constraint catalogue is defined. In contrast, IEEE-CIS, LCLD, and Sparkov show large protocol-dependent shifts, demonstrating that post-hoc filtering can alter not only metric values but also model-family rankings.

TABLE VIII
PARTIAL HOPSKIPJUMP CROSS-CHECK ON XGBOOST. HOPSKIPJUMP IS USED ONLY AS A DECISION-BASED BLACK-BOX CROSS-CHECK ON THE TREE MODEL. SEED COUNTS ARE SHOWN IN PARENTHESES WHERE FEWER THAN THREE SEEDS WERE RUN.

Dataset	HopSkipJump Robust PR-AUC
CCFD	0.002 ± 0.001
IEEE-CIS	0.060 ± 0.042 (2 seeds)
LCLD	0.185 (1 seed)
Sparkov	–

TABLE IX
PROTOCOL-INDUCED MODEL-FAMILY RANKING CHANGES UNDER SQUARE ATTACK

Dataset	Best A	Best B	Change
CCFD	XGBoost	XGBoost	No top change
IEEE-CIS	MLP	XGBoost	Top–bottom reversal
LCLD	XGBoost	Ensemble	Top change
Sparkov	XGBoost	XGBoost	Lower-rank change

The top and bottom rankings are reversed. On LCLD, the best model changes from XGBoost under Protocol A to the ensemble under Protocol B. On Sparkov, XGBoost remains first but the lower-rank ordering changes. Thus, protocol choice can affect model selection, not merely the numerical value of a robustness metric. Table VIII reports HopSkipJump as a partial decision-based cross-check on XGBoost only because of the computational cost. As its coverage is incomplete, HopSkipJump is not used for the main model-family ranking analysis. Square Attack remains the primary black-box attack for cross-model comparison.

D. Deployment-Aware Defences

Table X reports robust PR-AUC for the evaluated defence settings under the deployment protocol used for each dataset. CCFD uses Protocol A because no semantic constraints are

defined. IEEE-CIS, LCLD, and Sparkov use Protocol C2. The None, adversarial-training, and z-score input-validation columns are evaluated on the neural model, while the ensemble column is included as a cross-model comparison. This reporting choice aligns the defence comparison with the paper's central claim: robustness conclusions should be compared under the same deployment-aware protocol used for adversarial evaluation.

TABLE X
DEPLOYMENT-AWARE DEFENCE ROBUST PR-AUC WITH ENSEMBLE COMPARISON

Dataset	Prot.	None	Adv. Train	Z Input Val.	Ensemble
CCFD	A	0.598 ± 0.262	0.580 ± 0.225	0.572 ± 0.300	0.794 ± 0.074
IEEE-CIS	C2	0.412 ± 0.033	0.453 ± 0.023	0.367 ± 0.034	0.030 ± 0.008
LCLD	C2	0.105 ± 0.000	0.316 ± 0.004	0.105 ± 0.000	0.108 ± 0.001
Sparkov	C2	0.501 ± 0.033	0.577 ± 0.052	0.326 ± 0.013	0.093 ± 0.015

Adversarial training is the strongest neural-model defence on the constrained datasets under Protocol C2, improving robust PR-AUC on IEEE-CIS, LCLD, and Sparkov. On CCFD, where no semantic constraint catalogue is defined and Protocol A is used, the ranking is less stable. The ensemble obtains the highest mean robust PR-AUC, while adversarial training does not improve over the no-defence baseline. Z-score input validation provides no reliable robust-PR-AUC benefit under the tested setting and is consistently weaker than adversarial training on the constrained datasets. The ensemble column should be interpreted as a cross-model comparison rather than a neural-model defence: it is beneficial on CCFD but weak on the constrained datasets, especially IEEE-CIS and Sparkov.

VII. DISCUSSION

A. Protocol Sensitivity

The main empirical message is that robustness conclusions on financial fraud and credit-risk data are protocol-sensitive. Protocol B measures how many outputs of a particular unconstrained attack survive a feasibility filter. If very few survive, the attack may simply be inefficient at producing valid records. That does not imply that the model is safe against an attacker who respects constraints during generation.

This is clearest on LCLD. Protocol A and Protocol B leave only 3.7 feasible-flipped examples, while Protocol C2 produces 2,832.3 feasible-flipped examples under the same perturbation budget. The number of realistic successful attacks is therefore determined by the evaluation protocol, not only by the model. This supports the central FraudBench claim that fraud robustness must be evaluated with in-attack constraint integration, not only post-hoc filtering.

B. Feasibility and Capability Are Separate Axes

The IEEE-CIS results show that feasibility and attacker capability should not be collapsed into one metric. One-hot projection makes categorical records feasible and raises feasible-flipped counts from 0.0 to 93.7. However, adding a mutability mask reduces the feasible-flipped count to 4.3 because the realistic attacker loses access to many predictive dimensions. This is the opposite of LCLD, where adding a mask after projection increases the feasible-flipped count.

The design of asymmetry in FraudBench is meant to be useful rather than problematic, which shows that realistic fraud robustness depends on the overlap between mutable features and predictive features. If the attacker can modify features that the model relies on, as in LCLD, constraint-aware attacks remain dangerous. If the predictive signal lies in immutable or institution-controlled features, as in IEEE-CIS, capability constraints can substantially reduce attacker success.

C. Dual Reporting is Necessary

The experiments show that no single metric is sufficient. PR-AUC captures ranking degradation on imbalanced data, but does not indicate whether adversarial records are feasible. Feasibility captures constraint validity, but does not measure predictive degradation. Feasible-flipped count captures the intersection of both: the number of attacks that both fool the model and satisfy all constraints.

FraudBench therefore recommends reporting clean PR-AUC, robust PR-AUC, aggregate feasibility, feasible-flipped count, and filtered success rate together. The Square Attack results reinforce this need: several cells have very low robust PR-AUC under Protocol A but much higher Protocol-B robust PR-AUC after filtering. Without feasibility-aware metrics, these differences would be easy to misread as model robustness rather than attack filterability.

D. Relation to TabularBench

FraudBench should not be read as a replacement for TabularBench. TabularBench provides broad coverage of tabular deep learning architectures and constrained attack protocols [13]. FraudBench narrows the scope to financial fraud and credit-risk detection and studies the consequences of severe class imbalance, production-relevant tree models, domain-specific constraints, and feasibility-aware attacker success. The benchmark therefore answers a different application question: whether models used in fraud detection remain reliable under perturbations that are both adversarial and valid under financial-domain constraints.

The results show why this application-specific framing matters. Under Protocol B, post-hoc filtering can make an attack look weak because most generated records are invalid. Under Protocol C, the attacker generates valid records during the attack process. On LCLD, this distinction changes the feasible-flipped count from 3.7 to 2,832.3. This is not a small metric adjustment. It changes the interpretation of model safety.

VIII. LIMITATIONS AND RESPONSIBLE USE

FraudBench has several limitations. The main experiments use only three seeds, which may be insufficient for highly imbalanced datasets such as CCFD. The L_∞ budget in processed space is a standardised stress test rather than a monetary or attacker-cost-aware threat model. CAPGD is not directly applicable to XGBoost, so tree-model robustness should be

read mainly from black-box attacks such as Square Attack and HopSkipJump. Broader comparison with CAA and MOEVA, as well as more datasets and projection operators, remains future work. FraudBench is intended for defensive evaluation, model auditing, and benchmark comparison, not as operational guidance for committing fraud.

IX. Conclusion

This paper has presented FraudBench, a protocol-sensitive benchmark for adversarial robustness in tabular financial fraud and credit-risk detection. FraudBench combines public financial datasets, production-relevant model families, white-box and black-box attacks, PR-AUC-centred evaluation, and feasibility-aware attack metrics under matched evaluation protocols. The central finding is that fraud robustness is highly protocol-sensitive. Post-hoc filtering measures the filterability of unconstrained attacks, not the full success of a deployment-aware constrained attacker. On LCLD, under the tested CAPGD setting, integrating the instalment constraint and attacker mutability into attack generation raises feasible-flipped attacks from 3.7 under post-hoc filtering to 2,832.3 under projection plus mutability masking. The results on IEEE-CIS and Sparkov further show that one-hot validity is a recurring binding constraint, while Square Attack shows that protocol choice can change model-family conclusions.

Overall, financial fraud robustness evaluation should report predictive degradation and attack feasibility together. Domain constraints and attacker capability should be treated as part of attack generation, not merely as post-processing checks. Robust PR-AUC, aggregate feasibility, feasible-flipped count, and filtered success rate should be reported jointly.

References

[1] D. Lunghi, A. Simitsis, O. Caelen, and G. Bontempi, "Adversarial learning in real-world fraud detection: Challenges and perspectives," in *Proceedings of the Second ACM Data Economy Workshop (DEC)*. ACM, 2023.

[2] Nilson Report, "Payment card fraud losses worldwide," Issue 1298, Santa Barbara, CA, Jan. 2026, global payment card fraud losses were reported as $33.41 billion in 2024. [Online]. Available: https://nilsonreport.com/newsletters/1298/

[3] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," *arXiv preprint arXiv:1412.6572*, 2014.

[4] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, "Towards deep learning models resistant to adversarial attacks," in *International Conference on Learning Representations (ICLR)*, 2018.

[5] F. Croce and M. Hein, "Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks," in *International Conference on Machine Learning (ICML)*, 2020, pp. 2206–2216.

[6] A. Dal Pozzolo, O. Caelen, R. A. Johnson, and G. Bontempi, "Calibrating probability with undersampling for unbalanced classification," in *2015 IEEE Symposium Series on Computational Intelligence (SSCI)*. IEEE, 2015, pp. 159–166.

[7] T. Chen and C. Guestrin, "XGBoost: A scalable tree boosting system," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, 2016, pp. 785–794.

[8] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, and T.-Y. Liu, "LightGBM: A highly efficient gradient boosting decision tree," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2017, pp. 3149–3157.

[9] L. Grinsztajn, E. Oyallon, and G. Varoquaux, "Why do tree-based models still outperform deep learning on typical tabular data?" in *Advances in Neural Information Processing Systems (NeurIPS) Datasets and Benchmarks Track*, 2022.

[10] S. Ghamizi, M. Cordy, M. Gubri, M. Papadakis, A. Boystov, Y. Le Traon, and A. Goujon, "Search-based adversarial testing and improvement of constrained credit scoring systems," in *Proceedings of the 28th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering (ESEC/FSE)*, 2020, pp. 1089–1100.

[11] A. Chernikova and A. Oprea, "FENCE: Feasible evasion attacks on neural networks in constrained environments," *ACM Transactions on Privacy and Security*, vol. 25, no. 4, pp. 1–34, 2022.

[12] T. Simonetto, S. Dyrmishi, S. Ghamizi, M. Cordy, and Y. Le Traon, "A unified framework for adversarial attack and defense in constrained feature space," in *Proceedings of the 31st International Joint Conference on Artificial Intelligence (IJCAI)*, 2022, pp. 1313–1319.

[13] T. Simonetto, S. Ghamizi, and M. Cordy, "Constrained adaptive attack: Effective adversarial attack against deep neural networks for tabular data," *Advances in Neural Information Processing Systems*, vol. 37, pp. 27 817–27 849, 2024.

[14] F. Croce, M. Andriushchenko, V. Sehwag, E. Debenedetti, N. Flammarion, M. Chiang, P. Mittal, and M. Hein, "RobustBench: A standardized adversarial robustness benchmark," in *Advances in Neural Information Processing Systems (NeurIPS) Datasets and Benchmarks Track*, 2021.

[15] Y. Dong, Q.-A. Fu, X. Yang, T. Pang, H. Su, Z. Xiao, and J. Zhu, "Benchmarking adversarial robustness on image classification," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 321–331.

[16] T. Simonetto, S. Ghamizi, and M. Cordy, "TabularBench: Benchmarking adversarial robustness for tabular deep learning in real-world use-cases," in *Advances in Neural Information Processing Systems (NeurIPS) Datasets and Benchmarks Track*, 2024.

[17] IEEE Computational Intelligence Society and Vesta Corporation, "IEEE-CIS fraud detection," 2019, kaggle Competition Dataset, https://www.kaggle.com/c/ieee-fraud-detection.

[18] B. Harvard, "Sparkov data generation," GitHub repository, 2019, accessed: 2026-06-07. [Online]. Available: https://github.com/namebrandon/Sparkov_Data_Generation

[19] P. Grover, J. Xu, J. Tittelfitz, A. Cheng, Z. Li, J. Zablocki, J. Liu, and H. Zhou, "Fraud dataset benchmark and applications," *arXiv preprint arXiv:2208.14417*, 2022.

[20] M. Carminati, L. Santini, M. Polino, and S. Zanero, "Evasion attacks against banking fraud detection systems," in *23rd International Symposium on Research in Attacks, Intrusions and Defenses (RAID)*, 2020, pp. 285–300.

[21] K. El-Awady, "Adaptive stress testing for adversarial learning in a financial environment," *arXiv preprint arXiv:2107.03577*, 2021.

[22] F. Cartella, O. Anunciacao, Y. Funabiki, D. Yamaguchi, T. Akishita, and O. Elshocht, "Adversarial attacks for tabular data: Application to fraud detection and imbalanced data," *arXiv preprint arXiv:2101.08030*, 2021.

[23] J. L. Fok, Q. Zeng, S. Chen, O. Fawkes, and H. Chen, "Foe for fraud: Transferable adversarial attacks in credit card fraud detection," in *2025 IEEE International Conference on Web Services (ICWS)*. IEEE, 2025, pp. 286–292.

[24] A. Goldberg, G. Fanti, N. B. Shah, and Z. S. Wu, "Benchmarking fraud detectors on private graph data," in *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*. ACM, 2025.

[25] M. Andriushchenko, F. Croce, N. Flammarion, and M. Hein, "Square attack: A query-efficient black-box adversarial attack via random search," in *European Conference on Computer Vision (ECCV)*. Springer, 2020, pp. 484–501.

[26] J. Chen, M. I. Jordan, and M. J. Wainwright, "HopSkipJumpAttack: A query-efficient decision-based attack," in *2020 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2020, pp. 1277–1294.

[27] F. Tramèr, A. Kurakin, N. Papernot, I. Goodfellow, D. Boneh, and P. McDaniel, "Ensemble adversarial training: Attacks and defenses," in *International Conference on Learning Representations (ICLR)*, 2018.