Threat Domain Partitioning and Sorted Rejection Labeling: Benchmarking for Adversarial Environments

Charles Yeh, Daniel Lee, Hongkai Pan
charles@withpersona.com, daniel@withpersona.com, hongkai@withpersona.com

Persona

Abstract

Several distinctions make fraud detection different from other domains such that conventional machine learning classification metrics become impractical: adversarial adaptation, expensive labels, class imbalance, and unseen classes. These distinctions make many conventional classification metrics not only difficult to compute but even misleading. We present a practical framework that offers cheaper and more consistent benchmarks for such models. Our novel framework introduces two procedures: (1) ***threat domain partitioning****, which comprehensively defines the space of possible attacks into manageable categories, and (2)* ***sorted rejection labeling****, which efficiently measures model performance by focusing evaluation effort on the highest-risk cases. This framework replaces conventional classification metrics with alternatives that are directly comparable with business objectives such as user conversion and fraud exposure risk. Application of the framework on real-world fraud detection systems demonstrates significant reductions in labeling costs and much more consistent benchmarks while maintaining rigorous product standards, enabling rapid deployment cycles that match the pace of adversarial adaptation.*

1 Introduction

We work on a wide range of fraud detection models, including selfie liveness and ID verification. As AI driven fraud has grown more dynamic and sophisticated, manual labeling in these use cases has become more uncertain and costly to implement. This led us to build a new framework that addresses several key challenges unique to fraud detection.

1.1 Identifying metrics stable across performative drift

Fraud detection benchmarking in machine learning literature typically relies on conventional classification metrics such as precision, recall, F1-score, and AUC score for evaluation [1, 2, 3]. While some studies acknowledge the limitations of these metrics and supplement their analysis with additional measures [4, 5, 6], nearly all evaluations are conducted on static test sets that fail to account for performative drift. In the performative prediction framework, a model's predictions directly influence the environment:

when deployed, adversaries probe the model, observe outcomes, and adapt their tactics accordingly [7]. This adaptation causes conventional classification metrics to shift dramatically between pre-deployment and post-deployment measurements, rendering deployment decisions based on these metrics potentially misinformed. Consider the following example:

The flickering attack vector. A newly deployed model targeting stolen identity documents exhibited excellent performance in offline testing, achieving high precision, recall, F1-score, and AUC scores on a carefully curated test set. However, within hours of deployment, fraudsters realized this attack vector was no longer effective and ceased their attacks. This leads to a decrease in fraud caught or true positives (TP), in turn leading to sharp drops in all four conventional classification metrics: precision ($\frac{\text{TP}}{\text{TP+FP}}$), recall ($\frac{\text{TP}}{\text{TP+FN}}$), F1-score, and AUC score. But then due to the drop in metrics, the model is removed from production. However, when the model is removed, fraudsters who had been deterred by its presence return to exploit the now-undefended attack vector, causing fraud rates to spike. The "low performing" model was actually providing critical coverage that was invisible to the metrics.

Precision and recall measured around both deployment and rollback tell completely contradictory narratives and fail to capture the model's true operational value. Conventional classification metrics measured on static test sets fail to capture performative drift. This paper addresses these challenges by providing metrics that remain stable in dynamic environments and can therefore serve as clear and efficient decision criteria for deployment.

1.2 Improving labeling efficiency under class imbalance

In fraud detection scenarios, ground-truth labels may be scarce and expensive to obtain even though conventional classification metrics necessitate ground-truth labels for the entire test set. Furthermore, the rate of bad actors can range from 0.01% to 1%, depending on the use case [8, 1], implying that one might need to label 1,000 instances to obtain a single bad actor label. Ensuring sufficiently large samples for all classes is impractical in these scenarios, which motivates the need for procedures that operate more efficiently in the presence of such class imbalance.

1.3 Monitoring coverage rate across all attack vectors

Live fraud data is inherently sparse because fraudsters identify the most vulnerable attack vectors and exploit them at scale, meaning observed fraud data never covers the full range of possibilities. This necessitates a comprehensive taxonomy of potential attack vectors beyond what has been observed in the wild.

Identifying which attack vectors are most vulnerable at any given time is crucial for proactive defense and system prioritization. However, conventional classification metrics cannot be partitioned or aggregated across models, impeding our ability to track attack vector coverage over time. We require metrics that operate at the model level but also enable holistic analysis across all possible attack vectors.

2 Related work

These challenges in fraud detection mirror similar challenges in other adversarial contexts where attackers rapidly adapt their strategies. In contexts such as credit scoring, spam filtering, job application parsing, and intrusion detection, directly measuring a model's ability to distinguish between good and bad actors is both impractical due to labeling difficulties and misleading due to performative drift [9, 10, 8, 1, 11, 12, 13].

2.1 Adversarial adaptation in fraud detection

Recent work has explored incorporating performative drift into model training to optimize for post-drift performance in adversarial contexts. Strategic classification problems are typically modeled as leader-follower games, where the leader represents an institution deploying a classifier and the follower represents

an adaptive agent responding to the classification system [14, 15]. Credit scoring serves as the canonical example of this framework.

Fraud detection, however, presents a fundamentally different challenge. In our setting, not only do classification boundaries influence agent behavior, but the overall incidence rate and attack distribution shift as bad actors become deterred and cease activity altogether. This necessitates optimization over all possible attack vectors, including those not currently observed in the wild, distinguishing our adversarial context from traditional strategic classification contexts.

2.2 Ensemble approaches for adversarial robustness

Recent work has shown that dynamically weighted ensemble models—either through time-batched incremental learning or mixture-of-experts architectures—demonstrate improved adversarial robustness [16, 17]. The flexibility that comes from routing between sparse, specialized models is particularly relevant to fraud detection [18, 19]. Our framework adopts a mixture-of-experts approach where each expert model specializes in one or more threat domains. While prior work focuses on automated model generation and weighting, our expert models are handcrafted to leverage the many opportunities in feature engineering and data mining.

2.3 Class imbalance and ground truth constraints

Automated time-batched model generation approaches apply to fraud detection but face practical challenges from extreme class imbalance and sparse fraud representation. While some research addresses class imbalance during model training, many academic experiments rely on credit card fraud datasets that benefit from delayed but eventually reported ground truth [20]. Real-world fraud detection deployments, particularly those serving multiple clients, face fundamental ground truth challenges: limited access to post-deployment outcomes and inconsistent fraud definitions across use cases. These inconsistencies arise from mismatches between detection systems and use cases, such as the confounding of first-party and third-party fraud. These practical constraints motivate our focus on evaluation metrics that operate with limited labeling and remain stable across deployment contexts.

2.4 Evaluation metrics for fraud detection

The ISO/IEC 30107 standard defines metrics for biometric presentation attack detection, including Attack Presentation Classification Error Rate (APCER) and Bona fide Presentation Classification Error Rate (BPCER) [21]. While theoretically sound, these metrics remain expensive to compute in production environments as they require complete ground-truth labels for all classes. NIST's evaluation protocol employs APCER @ BPCER = 0.01, which aligns with our approach of setting a target false rejection rate [22]. We extend this framework by providing an efficient procedure for computing attack detection metrics to enable deployment decisions without exhaustive labeling.

2.5 Attack categorization frameworks

The ISO/IEC 30107 standard establishes a categorization framework for presentation attacks through Presentation Attack Instrument (PAI) species and points of attack [23]. Existing benchmark datasets have also proposed spoof type categorizations for fraud detection systems, such as CelebA-Spoof [24] for physical presentation attacks against face recognition systems. Our threat domain taxonomy extends these works by categorizing attacks based on both the material being presented and the method of attack.

Our taxonomy integrates multiple dimensions including attack entry points, physical presentation attacks, digital manipulation attacks, and hybrid approaches to provide a comprehensive framework with full coverage of all possible attack vectors in modern fraud detection systems. This multi-dimensional categorization enables the threat domain partitioning as well as the mixture of expert strategies central to our evaluation framework.

3 Framework overview

Our framework comprises two complementary procedures executed at distinct stages of the model lifecycle: threat domain partitioning, conducted prior to model development, and sorted rejection labeling, performed during deployment assessment.

3.1 Threat domain partitioning for bad actor coverage

Due to the dynamic and shifting nature of bad actors, we construct and use a representative set for evaluation rather than relying on sampling from the live population. This test set serves as a comprehensive taxonomy of observed attack vectors, and is continuously curated and partitioned into threat domain partitions at multiple levels of granularity. We regularly perform the following tasks to keep the taxonomy updated and comprehensive:

1. **Add observed fraud**: Classify newly observed fraud into new or existing threat domain partitions.
2. **Improve partitioning**: Refine and decompose existing partitions into more specific partitions as needed.
3. **Publish partitions**: Publish partition definitions for use in model development and evaluation.

All analysis and benchmarking centers on fraud capture rate (FCR), defined as the percentage of correctly identified bad actors. This metric is tracked both holistically across the threat domain and individually per threat domain partition. When designing a model, we perform the following tasks:

1. **Review partitions**: Review partitions for prioritization based on vulnerability.
2. **Design specialized model**: Select one or more vulnerable partitions and train a specialized model.
3. **Update metrics**: Upon model completion and deployment, update the relevant partition FCRs.

Threat domain partitioning and model development operate in two perpetual cycles that continuously inform and advance one another. This reframes the unseen classes problem into a problem space that can be solved by iteratively shipping models to systematically expand FCR over time, rather than every deployment being a careful balance between precision and recall.

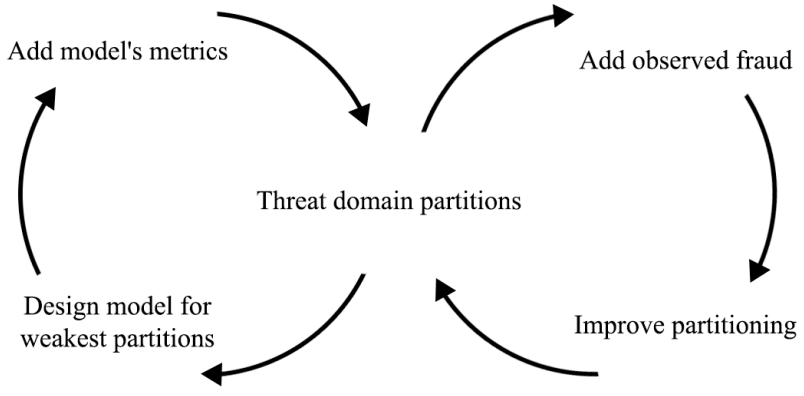

Figure 1: Threat domain partitioning refinement and model development cycles.

3.2 Sorted rejection labeling for good actor protection

Unlike bad actors, good actors do not actively adapt to deployed defenses and are therefore less affected by performative drift, allowing us to center evaluation around a predetermined false rejection rate (FRR) defined as $(\frac{\text{FP}}{\text{total}})$. In the following procedure, we evaluate the model on good actors sampled from the live population and bad actors sampled from the model's selected threat domain partitions:

1. Determine the target FRR, derived directly from business objectives.
2. Using good actor samples from the live population, identify the score threshold that comes as close as possible to the target FRR while remaining within it.

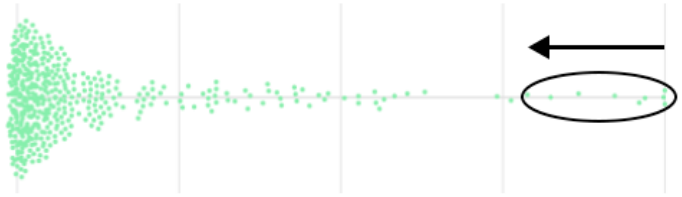

Figure 2: Labeling procedure from high to low scores.

3. Using bad actor samples from the threat domain partitions, calculate the FCR or the percentage of bad actors correctly rejected at that same score threshold.

Figure 3: Applying the determined threshold to the threat domain partitions.

This procedure inverts the conventional evaluation paradigm: given a predetermined constraint, we determine the corresponding model score threshold to evaluate the resulting FCR. Notably, using FRR as the primary constraint means that ground-truth labels are usually not required for the entire set, only for the highest risk instances until we reach the permissible false rejection count. In practice, this requires labeling only a small fraction of the dataset unless fraud rates are exceptionally high. To account for operational maintenance costs, deployment requires models to achieve a minimum FCR threshold. Models meeting this requirement within the target FRR constraint can proceed to production.

4 A case study: selfie liveness verification

We demonstrate the application of our framework through selfie liveness verification, a widely deployed identity verification mechanism that requires users to capture real-time photographs or video to establish physical presence and verify claimed identity. With this use case, we implement both components: threat domain partitioning and sorted rejection labeling.

4.1 Threat domain partitioning

The attack surface for selfie liveness verification is expansive and rapidly evolving, particularly with recent advances in deepfake and generative AI technologies. Our threat domain taxonomy was initially constructed from broad attack categories and iteratively refined through continuous observation of emerging fraud patterns. The top-level partitions comprise:

- **Generative AI**: Synthetically generated images from generative models designed to mimic authentic selfies.
- **Digital tampering**: Manipulation of genuine images through deepfakes, inpainting, or outpainting techniques to alter identity or appearance.
- **Digital render**: Computer-generated avatars, 3D models, or video game screenshots submitted as identity proof.
- **Replay**: Pre-recorded images or videos presented to circumvent liveness detection mechanisms.
- **Physical replica**: Physical artifacts such as silicone masks or printed photographs used for impersonation.
- **Evasion**: Deliberate modification of physical attributes through makeup, face paint, or occlusion to evade detection.

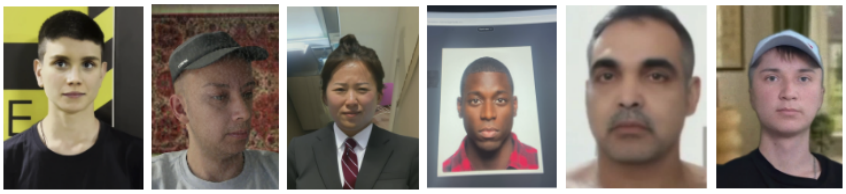

Figure 4: Example threat domain partitions for selfie liveness verification.

Generative AI represents the most dynamic partition within our threat domain taxonomy, reflecting both the rapid evolution of generative models and the increasing accessibility of AI-based attack tools. This partition is hierarchically subdivided by underlying model architecture with unknown or unclassified instances temporarily assigned to a residual category pending further analysis. Representative subpartitions within the Generative AI domain include:

- **GAN-based models**: Generative Adversarial Network architectures
  - StyleGAN
  - ProGAN
- **Diffusion-based models**: Denoising diffusion probabilistic models
  - Stable Diffusion
  - Midjourney
  - DALL-E
  - Gemini
  - Sora

Note that individual fraud instances may exhibit characteristics spanning multiple partitions simultaneously, as adversaries deliberately combine techniques to obfuscate their methods and evade detection across multiple defensive layers. For example, a sophisticated attack comprising a pre-recorded video with deepfake manipulation, virtual background substitution, and simulated PRNU artifacts would include all of the following partitions:

- *Replay* → *Pre-recorded video*
- *Digital tampering* → *Deepfake*
- *Digital tampering* → *Virtual background*
- *Digital tampering* → *Simulated PRNU*
- *Generative AI* → *Diffusion-based models* → *Stable Diffusion*
- *Evasion* → *Headwear* → *Wig*

For model development, we identify and target one or more vulnerable partitions based on observed fraud trends and coverage gaps. Critically, selected partitions need not be adjacent within the taxonomy hierarchy nor reside at uniform granularity levels. For instance, a diffusion-detection model might simultaneously address *Digital tampering* → *Inpainting* → *Photoshop* alongside all sub-partitions within *Generative AI* → *Diffusion-based models*, spanning multiple branches and hierarchical depths. Once the target threat domain partitions are identified, we proceed to model development and training. After model training is complete, we proceed to the second phase of the framework: sorted rejection labeling.

4.2 Sorted rejection labeling

To set up the evaluation, we establish target FRRs derived from business requirements and operational constraints. For selfie liveness verification, we employ three tolerance levels, each with corresponding response policies:

- **FRR = 0.01%**: With minimal false rejections, rejections can be fully automated with negligible user friction.
- **FRR = 0.1%**: With moderate false rejections, fully automated rejection is still acceptable but only recommended for risk-sensitive contexts.
- **FRR = 1.0%**: With substantial false rejections, escalation to secondary verification or manual review is recommended.

Sort model results by risk. The model is evaluated on good actors sampled from the live population and bad actors sampled from the targeted threat domain partitions. All instances are ranked by model score in descending order of predicted risk. We then determine the maximum number of false rejections permissible under the target FRR constraint based on the population size.

Label from highest to lowest risk. Manual review begins with the highest-risk instance and proceeds monotonically toward the lowest-risk instance until the cumulative count of labeled false rejections matches the permissible false rejection count. For example, at a 0.1% FRR in a population of 1 million, we label instances until 1,000 false rejections are observed. While the upper bound for required labels is 1 million, we would only approach that number if the dataset was nearly all fraudulent. In practice, we typically label only a few thousand instances because fraud is rare.

Set model score threshold. The score of the final permissible false rejection establishes the model score threshold for the specified FRR level. All instances scoring at or above this threshold are classified as high-risk and rejected.

Calculate fraud capture rate. Using the established threshold, we evaluate the model against bad actors from the targeted threat domain partitions. The FCR represents the percentage of fraudulent instances correctly identified at the threshold, providing a direct measure of the model's effectiveness within the FRR constraint.

Deployment criteria. The model is deployed if the FCR meets or exceeds the deployment threshold. For the three target FRRs, we typically use the following deployment thresholds:

- **FRR = 0.01%**: FCR $\geq 40\%$

- **FRR = 0.1%**: FCR $\geq 60\%$
- **FRR = 1.0%**: FCR $\geq 80\%$

These thresholds reflect practical operational constraints: the 0.01% FRR level accepts lower coverage in exchange for minimal user friction, while higher FRR levels demand near-complete threat detection to justify the increased operational impact of false rejections.

The full path for sorted rejection labeling proceeds as follows: Target FRR $\rightarrow$ Permissible false rejections $\rightarrow$ Model score threshold $\rightarrow$ FCR $\rightarrow$ Deployment decision.

5 Discussion

5.1 Truth-indeterminate data

The flexibility of false rejection allows us to handle truth-indeterminate data better than metrics that are tied to the model's designed purpose, such as false positives.

False rejection vs. false positive. False rejection is more flexible than false positive because it is not tied to the model's designed purpose, but rather to the business objective of the system. For example, if a deepfake detection model flags a physical spoof, it would conventionally be classified as a false positive because the prediction does not match the model's specific purpose. In our framework, this would not be a false rejection because the physical spoof is indeed indicative of fraud and should be rejected. Furthermore, false rejection rate ($\frac{\text{FP}}{\text{total}}$) is cheaper to compute than false positive rate ($\frac{\text{FP}}{\text{FP+TN}}$) because finding the true negative count is very expensive in datasets with scarce labels and extreme class imbalance.

Fraud or not fraud? We often observe data that is difficult to reliably label due to situations that produce low image quality, such as dark lighting or lower-end cameras. We call these adverse scenarios, where special conditions cause data from one class to appear to be in the other. Legitimate users may accidentally encounter these scenarios, while bad actors intentionally exploit them to evade detection. Because users can make multiple attempts in our verification flows, we treat low-confidence instances as user error and reject them. This allows legitimate users to retry with better conditions, while sequence modeling detects bad actors who repeatedly exploit adverse scenarios. Resubmission typically yields higher quality data that the model is more likely to classify correctly as low risk.

5.2 Compounding of false rejection rate

Because we develop models in parallel and benchmark them individually on false rejection rate, the overall system false rejection rate increases with each model deployment. This occurs because each model's false rejections do not overlap perfectly with those of other models. Ideally, we would evaluate each new model as an ensemble combined with all existing models to precisely measure the overall system's false rejection rate. However, for practical reasons, we instead tolerate temporarily elevated false rejection rates and then periodically recalibrate the ensemble model back to the target false rejection rate through sorted rejection labeling.

5.3 Generalization to other performative models

While we demonstrate this framework in the context of fraud detection in selfie liveness verification, the core principles extend to any strategic classification context with unknown repeated adversaries. This circumstance is common in scenarios with a larger general population of legitimate actors and a smaller set of malicious actors, such as spam filtering, malware detection, content moderation, and intrusion detection.

6 Future directions

6.1 Standardization of fraud threat domains

There is a pressing need to establish standardized definitions and taxonomies that comprehensively span the entire landscape of fraud threat domains. Although some datasets and taxonomies exist for particular fraud categories [24, 25], no unified resource currently captures the full range of fraud types, including their combinations and permutations. Since the effectiveness of anti-fraud systems is determined by their weakest link, developing a unified threat taxonomy would enable repeatable evaluation and systematic iteration of deployed systems.

We currently implement this framework internally for several use cases, including selfie liveness and government ID verification. While we are still refining and formalizing our threat domain taxonomies, we aim to contribute them to the research community at a future date. Establishing common taxonomies would facilitate more effective knowledge sharing and enable systematic comparison of defensive strategies across organizations and use cases.

6.2 Utilizing medium-risk predicted scores

Because sorted rejection labeling concentrates scrutiny on the highest-risk cases, sophisticated adversaries adapt by intentionally crafting attacks that yield moderate risk scores—high enough that they aren't classified as benign, but low enough to escape the focused evaluation at the risk tail. This strategic targeting of the "middle" risk region allows attackers to evade automated detection, exploiting the evaluation framework's emphasis on the extremes. We are exploring methods for countering this evasion tactic through the combination and aggregation of many weaker signals to identify anomalies. By ensembling or cross-referencing model outputs, defenders can expose adversarial behaviors that manifest only as weak indicators within individual partitions but become strong indicators when viewed collectively.

However, while this multi-model synthesis can lead to more precise and robust detection, it introduces substantial operational complexity. Maintaining such interconnected systems requires continuous calibration of score normalization, monitoring for drift across models, and managing dependency chains between model updates.

7 Conclusion

Conventional machine learning classification metrics fail in adversarial environments where intelligent attackers rapidly adapt their strategies. This paper introduces a framework that fundamentally changes how we evaluate and deploy models in such practical environments. Our framework centers on two key innovations: threat domain partitioning, which comprehensively organizes the space of possible attacks into manageable categories, and sorted rejection labeling, which efficiently measures system performance by focusing evaluation effort on the highest-risk cases.

Threat domain partitioning's core strength lies in its ability to identify and highlight vulnerable attack vectors across the entire possible attack space. This capability is essential for ensuring impactful model development and enables parallel model design, where separate teams may train and evaluate models in isolated environments. Crucially, the results and progress from each independent team fit together within a cohesive, overarching framework: the collective outputs from all teams combine to provide comprehensive coverage of the entire threat landscape.

Sorted rejection labeling's key advantage is its ability to target and control the false rejection rate (FRR), grounding the overall framework in a common business objective. It enables efficient model evaluation with metrics that remain stable in the face of performative drift. By explicitly setting thresholds that correspond to concrete FRR values such as 0.01% or 0.1%, organizations can predetermine the exact percentage of users who may experience friction or intervention due to the deployed system.

Together, they form our framework for consistent, repeatable, and scalable model development and deployment in the face of rapid adversarial adaptation.

8 Acknowledgments

We thank Jinxing Li and Injee Jeong for their review, comments, and suggestions on the draft, as well as James Chang and Daniel George for their continued input on the framework's implementation and extensions.

References

[1] 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.

[2] Z. Yan, Y. Zhang, X. Yuan, S. Lyu, and B. Wu, "DeepfakeBench: A comprehensive benchmark of deepfake detection," in *Proceedings of the 37th Conference on Neural Information Processing Systems (Datasets and Benchmarks)*, 2023.

[3] A. Dal Pozzolo, G. Boracchi, O. Caelen, C. Alippi, and G. Bontempi, "Credit card fraud detection: A realistic modeling and a novel learning strategy," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 29, no. 8, pp. 3896–3912, 2018.

[4] B. Dolhansky, J. Bitton, B. Pflaum, J. Lu, R. Howes, M. Wang, and C. Canton Ferrer, "The DeepFake Detection Challenge (DFDC) dataset," *arXiv preprint* arXiv:2006.07397, 2020.

[5] L. Jiang, Z. Guo, W. Wu, Z. Liu, Z. Liu, and C. C. Loy, "DeeperForensics Challenge 2020 on real-world face forgery detection: Methods and results," *arXiv preprint* arXiv:2102.09471, 2021.

[6] I. Chingovska, A. Anjos, and S. Marcel, "On the effectiveness of local binary patterns in face anti-spoofing," in *BIOSIG—Proceedings of the International Conference of the Biometrics Special Interest Group*, 2012.

[7] J. C. Perdomo, T. Zrnic, C. Mendler-Dünner, and M. Hardt, "Performative prediction," in *Proceedings of the 37th International Conference on Machine Learning*, vol. 119, pp. 7599–7609, 2020.

[8] Y.-A. Lucas, P. Joly, O. Dufour, T. Pichon, and A. Braud, "Credit card fraud detection using machine learning: A survey," *arXiv preprint* arXiv:2010.06479, 2020.

[9] N. Dalvi, P. Domingos, S. Sanghai, and D. Verma, "Adversarial classification," in *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 99–108, 2004.

[10] M. Brückner, C. Kanzow, and T. Scheffer, "Static prediction games for adversarial learning problems," *Journal of Machine Learning Research*, vol. 13, pp. 2617–2654, 2012.

[11] A. A. Abdallah, M. A. Maarof, and A. Zainal, "Fraud detection system: A survey," *Journal of Network and Computer Applications*, vol. 68, pp. 90–113, 2016.

[12] Y. Chen, Y. Liu, and C. Podimata, "Learning strategy-aware linear classifiers," in *Advances in Neural Information Processing Systems*, vol. 33, pp. 15265–15276, 2020.

[13] G. Brown, S. Hod, and I. Kalemaj, "Performative prediction in a stateful world," in *Proceedings of the International Conference on Artificial Intelligence and Statistics*, pp. 6045–6061, 2022.

[14] M. Hardt, N. Megiddo, C. H. Papadimitriou, and M. Wootters, "Strategic classification," in *Proceedings of the ACM Conference on Innovations in Theoretical Computer Science*, pp. 111–122, 2016.

[15] T. Zrnic, E. Mazumdar, S. S. Sastry, and M. I. Jordan, "Who leads and who follows in strategic classification?" in *Advances in Neural Information Processing Systems*, vol. 34, pp. 15257–15269, 2021.

[16] J. Z. Kolter and M. A. Maloof, "Dynamic weighted majority: An ensemble method for drifting concepts," *Journal of Machine Learning Research*, vol. 8, pp. 2755–2790, 2007.

[17] J. Puigcerver, R. Jenatton, C. Riquelme, P. Awasthi, and S. Bhojanapalli, "On the adversarial robustness of mixture of experts," in *Advances in Neural Information Processing Systems*, 2022.

[18] R. J. Elwell and R. Polikar, "Incremental learning of concept drift in nonstationary environments," *IEEE Transactions on Neural Networks*, vol. 22, no. 10, pp. 1517–1531, 2011.

[19] M. Aspis, S. A. Cajas Ordónez, A. L. Suárez-Cetrulo, and R. Simón Carbajo, "DriftMoE: A mixture of experts approach to handle concept drifts," *arXiv preprint* arXiv:2507.18464, 2025.

[20] P. K. Chan, W. Fan, A. L. Prodromidis, and S. J. Stolfo, "Distributed data mining in credit card fraud detection," *IEEE Intelligent Systems*, vol. 14, no. 6, pp. 67–74, 1999.

[21] ISO/IEC 30107-3:2023, *Information technology—Biometric presentation attack detection—Part 3: Testing and reporting*, 2023. [Online]. Available: `https://www.iso.org/standard/79520.html`

[22] M. Ngan, P. Grother, and A. Hom, "Face Analysis Technology Evaluation (FATE) Part 10: Performance of passive, software-based presentation attack detection (PAD) algorithms," NIST Interagency/Internal Report 8491, 2023. [Online]. Available: `https://nvlpubs.nist.gov/nistpubs/ir/2023/NIST.IR.8491.pdf`

[23] ISO/IEC 30107-1:2023, *Information technology—Biometric presentation attack detection—Part 1: Framework*, 2023. [Online]. Available: `https://www.iso.org/standard/83828.html`

[24] Y. Zhang, Z. Yin, J. Shao, Z. Liu, S. Yang, Y. Xiong, W. Xia, Y. Xu, M. Luo, J. Liu, J. Li, Z. Chen, M. Guo, H. Li, J. Liu, P. Gao, T. Hong, H. Han, S. Liu, X. Chen, D. Qiu, C. Zhen, D. Liang, Y. Jin, and Z. Hao, "CelebA-Spoof Challenge 2020 on face anti-spoofing: Methods and results," *arXiv preprint* arXiv:2102.12642, 2021.

[25] Y. Ju, S. Jia, J. Cai, H. Guan, and S. Lyu, "GLFF: Global and local feature fusion for AI-synthesized image detection," *IEEE Transactions on Multimedia*, vol. 26, pp. 4073–4085, 2023.