Active Learning for Credit Card Fraud Detection in Imbalanced Transaction Data

Noor Shahin 326396256 **Kobi Amit** 206107344 **Noam Alter** 318306792 **Elad Polak** 205818818

{noor.shahin, kobiamit, noam.alter, eladpollak}@campus.technion.ac.il

Technion – Israel Institute of Technology
Data Analysis and Presentation Lab (096260)

Git Repository: **Active Learning for Credit Card Fraud Detection**

Abstract

Credit card fraud detection poses a highly imbalanced classification challenge in which fraudulent transactions represent less than 0.2% of all activity. Because expert annotation is expensive, improving label efficiency is essential. We evaluate nine active learning (AL) strategies (including random, uncertainty-based, committee-based, cost-balanced, fraud-aware, and two hybrid methods) on the public Kaggle credit card fraud dataset (284,807 transactions, 492 frauds). Our main methodological contributions are two novel domain-tailored hybrid strategies: *FRaUD++ Hybrid*, which mixes fraud-aware scoring with Query-by-Committee (QBC) disagreement, and *GraphHybrid*, which augments FRaUD++ with graph-based hub and bridge signals before combining with QBC. Across three random seeds and multiple labeling budgets, both hybrid methods achieve substantially higher fraud discovery than simple AL baselines. At 5000 labels, *GraphHybrid* recovers roughly **61% more frauds than QBC**, a standard committee-based baseline, and up to **296% more than random sampling**, while attaining strong final metrics (median AUPRC 0.826; median AUROC 0.961). These findings demonstrate that integrating fraud-aware priors, rarity cues, structural context, and committee disagreement yields markedly more label-efficient fraud detection under extreme imbalance.

Keywords: AL; fraud detection; class imbalance; graph-based learning; LightGBM

Introduction

Credit card fraud detection remains one of the most challenging and high-stakes problems in modern financial systems. Institutions process millions of transactions daily, yet only a tiny fraction (492 out of 284,807 in our dataset; approximately 0.17%) are fraudulent. This extreme imbalance limits the effectiveness of supervised machine learning models, as the majority class dominates the training signal. In addition, accurate fraud labeling requires expert human verification and is therefore costly and slow. As fraud patterns evolve over time, new labeled examples are needed for continuous model updates, but randomly selected samples are almost always legitimate, making random annotation highly inefficient.

AL provides a principled framework for improving label efficiency by iteratively selecting the most informative points from an unlabeled pool. Classical strategies such as entropy sampling, margin sampling, and QBC can reduce labeling demands by prioritizing ambiguous samples. However, these approaches were developed primarily for more balanced domains. In highly imbalanced, cost-sensitive settings like fraud detection, uncertainty-based sampling often selects majority-class outliers, exhibits unstable behavior across seeds, and fails to reliably discover frauds early in the labeling process.

To address these limitations, we propose and evaluate a suite of nine AL strategies, including five standard baselines and four fraud-aware or hybrid methods. Our primary methodological contributions are two novel domain-tailored hybrid approaches: (1) *FRaUD++ Hybrid*, which combines a fraud-aware score (built from model fraud probability, focal prior, boundary proximity, and rarity cues) with QBC disagreement; and (2) *GraphHybrid*, which augments FRaUD++ with structural information derived from a k-nearest-neighbor similarity graph over high-risk transactions, incorporating both hub and bridge signals before mixing with QBC. These hybrid strategies jointly exploit fraud likelihood, uncertainty, structural diversity, and committee disagreement to balance exploration and exploitation throughout the AL loop.

The central goal of this project is to determine whether such hybrid, structure-aware strategies can improve label efficiency and accelerate fraud discovery under fixed annotation budgets. Across three random seeds and multiple budgets, *GraphHybrid* and *FRaUD++ Hybrid* achieve substantially higher fraud discovery than simple AL baselines. At 5000 labels, the hybrid methods recover approximately **61% more frauds than QBC**, a standard committee-based baseline, and up to **296% more than random sampling**, while achieving strong end-of-budget metrics (median AUPRC 0.826; median AUROC 0.961). Although certain fraud-aware baselines remain competitive in AUROC or final fraud counts, the hybrid methods consistently offer superior stability and label efficiency across seeds and budgets.

In summary, this work contributes:

- A modular and reproducible AL framework for fraud detection, supporting multiple strategies, seeds, and evaluation metrics.
- Two novel hybrid sampling algorithms: **FRaUD++ Hybrid**, which combines fraud-aware scoring with committee-based disagreement, and **GraphHybrid**, which augments fraud-aware scoring with graph-based structural signals and committee-based disagreement.
- A comprehensive empirical evaluation on the public Kaggle credit card fraud dataset, analyzing label efficiency, early fraud discovery, and cross-seed stability.

The remainder of this report is organized as follows: Section reviews prior work on AL and imbalanced classification. Section introduces the proposed algorithms. Section details

the dataset, baselines, and evaluation metrics. Section presents the experimental results. Section concludes with implications and directions for future work.

Git Repository: **Active Learning for Credit Card Fraud Detection**

Related Work

AL has long been proposed as a way to reduce annotation costs by selecting informative unlabeled samples. Early foundational methods include uncertainty sampling (Lewis & Gale, 1994), which queries instances with the lowest classifier confidence, and QBC (Seung, Opper, & Sompolinsky, 1992), which measures disagreement across an ensemble. While effective in balanced text classification settings, these classical approaches tend to perform poorly under extreme class imbalance. In such cases, uncertainty is often dominated by majority-class outliers, causing the AL loop to over-query non-fraudulent examples and delay the discovery of meaningful minority samples.

In fraud detection, where the positive class may represent less than 0.2% of the data, several studies have emphasized the need for imbalance-aware querying. Dal Pozzolo et al. (Dal Pozzolo, Caelen, Le Borgne, Waterschoot, & Bontempi, 2015) demonstrated that standard AL strategies fail to target fraudulent instances efficiently, and showed that cost-sensitive adjustments and calibrated probability estimates can significantly improve minority-class recall. Their findings highlight a central limitation of classical AL: uncertainty scores alone do not reliably correlate with fraud likelihood.

A complementary line of work focuses on enhancing AL through diversity and representativeness. Sener and Savarese (Sener & Savarese, 2018) formulated core-set selection as a geometric coverage problem, ensuring that queried points adequately represent the overall data distribution. Ash et al. (Ash, Zhang, Krishnamurthy, Langford, & Agarwal, 2020) extended this idea with BADGE, combining gradient-based uncertainty with feature-space diversity to improve batch selection. These methods show that diversity helps avoid redundancy and encourages sampling from rare regions of the feature space, a desirable property in fraud detection where minority cases cluster sparsely.

Structure-aware methods further leverage relational information between samples. Graph-based learning has been widely explored in semi-supervised and active settings, with surveys such as Wu et al. (Wu et al., 2020) illustrating how k-nearest-neighbor graphs support label propagation and manifold-aware uncertainty. Such graph representations capture local density, hubs, and boundary structures (patterns that often align with fraud behaviors), which tend to form tight, high-risk clusters in embedding space.

Taken together, prior research suggests that: (1) classical AL is insufficient under severe imbalance; (2) cost-sensitive and calibrated approaches improve minority detection; and (3) diversity and graph structure help uncover underrepresented regions in complex datasets. Our work builds directly upon these insights by integrating uncertainty, cost awareness, and graph-guided diversity into hybrid sampling algorithms tailored specifically for rare-event fraud detection.

Preliminaries and Problem Definition

We study the problem of **AL for credit card fraud detection** under extreme class imbalance. Each transaction is represented as a feature vector $\mathbf{x}_i \in \mathbb{R}^d$ with an associated binary label $y_i \in \{0,1\}$, where $y_i = 1$ denotes fraud. The dataset used in this project contains 284,807 transactions with 492 fraud cases (0.17%), making the minority class exceedingly rare. Because fraud labeling requires expert verification, annotation is **costly**, **slow**, and cannot be applied to the entire dataset, motivating an AL approach.

Notation

Let the dataset (D) be decomposed into a labeled (L) and unlabeled (U) portion:

$$\mathcal{D} = \mathcal{L} \cup \mathcal{U}, \qquad \mathcal{L} \cap \mathcal{U} = \emptyset.$$

At iteration t of the AL loop:

- $\mathcal{L}_t$: labeled set available for training,
- $\mathcal{U}_t$: unlabeled pool,
- f_{θ_t}: classifier (LightGBM) trained on $\mathcal{L}_t$ with parameters θ_t,
- $p_{\theta_t}(y = 1 \mid \mathbf{x})$: predicted fraud probability,
- Q_t: acquisition function that assigns a score $Q_t(\mathbf{x})$ to each $\mathbf{x} \in \mathcal{U}_t$,
- B_t: batch of samples selected for labeling at iteration t,
- B: total labeling budget (fixed across the entire AL run).

The acquisition step selects a batch of points with the highest informativeness:

$$B_t = \arg\max_{\mathcal{B} \subseteq \mathcal{U}_t} \sum_{\mathbf{x} \in \mathcal{B}} Q_t(\mathbf{x}) \quad \text{s.t.} \quad |\mathcal{B}| \leq b_t,$$

where b_t is the per-iteration batch size (constant in our experiments).

After labeling B_t, the sets update as:

$$\mathcal{L}_{t+1} = \mathcal{L}_t \cup \{(\mathbf{x}, y) : \mathbf{x} \in B_t\}, \qquad \mathcal{U}_{t+1} = \mathcal{U}_t \setminus B_t.$$

Problem Definition

The core problem addressed in this project is:

> *Given a highly imbalanced dataset and a fixed labeling budget, select a sequence of unlabeled instances whose annotation maximizes fraud discovery and improves downstream model performance.*

More formally, the objective of AL in this context is:

$$\max_{Q_1,\ldots,Q_T} \text{Perf}(f_{\theta_T}; \mathcal{L}_T) \quad \text{s.t.} \quad \sum_{t=1}^{T} |B_t| \leq B,$$

where $\text{Perf}(\cdot)$ is evaluated using metrics appropriate for rare-event detection:

- Area Under the Precision-Recall Curve (AUPRC)
- Area Under the ROC Curve (AUROC)
- Recall at a fixed false positive rate (recall@0.1% FPR)
- Profit-based metrics reflecting cost asymmetry
- Number of frauds discovered as a function of labels

Thus, the goal is not merely classification accuracy, but **label efficiency**: achieving strong detection performance and early fraud discovery using as few labeled transactions as possible.

Dataset Summary

We use the public Kaggle Credit Card Fraud Detection dataset (Kaggle, 2018), containing standardized V-features extracted via PCA, along with Time, Amount, and the binary fraud label. The class distribution is:

Class	Count	Percentage
Legitimate	284,315	99.83%
Fraudulent	492	0.17%

This extreme imbalance makes random labeling highly inefficient, and motivates the design of fraud- and structure-aware acquisition functions.

Methodology

System Overview

Our system follows a standard pool-based AL loop adapted to highly imbalanced fraud detection. At iteration t, the classifier f_{θ_t} is trained on the labeled set $\mathcal{L}_t$, each $\mathbf{x} \in \mathcal{U}_t$ is assigned an acquisition score $Q_t(\mathbf{x})$, and a batch B_t of the top-scoring samples is selected and annotated. The newly labeled batch is then added to form $\mathcal{L}_{t+1}$ and removed from $\mathcal{U}_t$ to obtain $\mathcal{U}_{t+1}$. This process repeats until the labeling budget B is exhausted. Throughout the loop, we track fraud discovery, AUPRC, AUROC, precision@0.1% FPR, and profit.

The goal of our methodology is to design acquisition functions that (1) maximize early fraud discovery, (2) remain stable across seeds, and (3) exploit structural patterns within high-risk transactions.

Base Model

We use **LightGBM** (Ke et al., 2017) as the classifier f_θ due to its efficiency, ability to model non-linear interactions, robustness to heterogeneous feature scales, and strong performance on tabular financial datasets. The model outputs fraud probability scores:

$$p_\theta(y = 1 \mid \mathbf{x}) \in [0, 1].$$

In the experiments reported here we rely on LightGBM's native probabilities without an additional post-hoc calibration step (`use_calibration = false` in our configuration), while explicitly evaluating metrics that target the low-FPR regime (e.g., recall at 0.1% FPR). To handle the extreme class imbalance in the labeled set, we dynamically set LightGBM's `scale_pos_weight` parameter at each AL iteration based on the ratio of negative to positive examples in the current labeled set $\mathcal{L}_t$, so that minority-class (fraud) instances receive a higher effective weight during training.

The per-iteration batch size b_t is set to a fixed value in the configuration (400 samples in our experiments), although the final batch in a run may be smaller if fewer than b_t labels remain under the total labeling budget B or in the unlabeled pool. The labeled pool grows as the AL loop progresses.

Baseline Acquisition Strategies

We implement five commonly used AL baselines to serve as reference points:

1. Random sampling: Selects b_t unlabeled samples uniformly at random. This provides a lower bound on AL performance and is important for evaluating relative label-efficiency gains.

2. Entropy sampling (Lewis & Gale, 1994):

$$\mathrm{ENT}(\mathbf{x}) = -p(\mathbf{x}) \log p(\mathbf{x}) - (1 - p(\mathbf{x})) \log(1 - p(\mathbf{x})).$$

3. Margin sampling:

$$\mathrm{MAR}(\mathbf{x}) = 1 - |2p(\mathbf{x}) - 1|.$$

Scores are highest near the decision boundary.

4. Query-by-Committee (QBC) (Seung et al., 1992): A committee of heterogeneous models (Logistic Regression, Random Forest, and an MLP) is trained on the labeled data. Acquisition is proportional to committee disagreement, which is calculated using Jensen–Shannon divergence over the models' probability estimates.

5. Cost-balanced entropy: Reweights uncertainty to overemphasize potential positives:

$$\mathrm{CB}(\mathbf{x}) = w^+(p(\mathbf{x}))\,\mathrm{ENT}(\mathbf{x}),$$

where w^+ is a class-cost prior reflecting fraud rarity.

These baselines constitute standard choices but perform poorly under extreme imbalance, often prioritizing majority-class outliers.

Fraud-Aware Strategies

This section describes the fraud-aware strategies implemented in `strategies.py`. The strategies form a family: we start from a FRaUD base score built from model uncertainty, prior-adjusted fraud probability, and boundary proximity, and then extend it with rarity-aware and hybrid components.

Fraud Score (Fraud). This strategy (`FRaUDSampler`, reported as `fraud` in our results), computes a base score that combines model uncertainty ($U(\mathbf{x})$, i.e., entropy), a focal prior term that upweights instances with $p(\mathbf{x})$ above an effective fraud prior, and Threshold-Adaptive Boundary Proximity $\widetilde{B}(\mathbf{x})$. The score is:

$$q_{\mathrm{FRaUD}}(\mathbf{x}) = \big[U(\mathbf{x})\big]^{\alpha} \cdot \left(\frac{p(\mathbf{x})}{\mathrm{prior} + p(\mathbf{x})}\right)^{\gamma} \cdot \Big((1-\beta_t) + \beta_t\,\widetilde{B}(\mathbf{x})\Big), \tag{1}$$

where α and γ are fixed hyperparameters controlling the influence of uncertainty and the focal prior term, and β_t is a scheduled weight for boundary proximity. Here $U(\mathbf{x})$ is the entropy of $p(\mathbf{x})$, prior denotes an effective fraud prior (the maximum of the configured global prior and the observed fraud rate in the labeled set), and $\widetilde{B}(\mathbf{x})$ is a normalized boundary score that emphasizes transactions near the operating threshold. This is implemented via the `_base_score` and `_mix_tabp` helper functions. This score is used as the `fraud` baseline and as a building block for FRaUD++.

Fraud++ Score (FRaUD++). This strategy (`FRaUDPlusSampler`, reported as `fraudpp` in our results) enhances the FRaUD score by multiplying it with a local rarity boost $R(\mathbf{x})$. The rarity term is designed to assign higher scores to high-risk samples in sparse, underrepresented regions (i.e., "novel" fraud modes). The score is:

$$q_{\text{FRaUD++}}(\mathbf{x}) = q_{\text{FRaUD}}(\mathbf{x}) \cdot \big(1 + \lambda_{R,t}\, R(\mathbf{x})\big), \tag{2}$$

where $\lambda_{R,t}$ is a time-dependent weight controlling the strength of the rarity correction. This score is computed by the `_rarity_boost` function, which is applied on top of the `_base_score`. In practice, $R(\mathbf{x})$ is computed and applied only for a high-risk candidate subset consisting of the top-scoring points under q_{FRaUD}; for all remaining pool points we set $R(\mathbf{x}) = 0$, so that $q_{\text{FRaUD++}}(\mathbf{x}) = q_{\text{FRaUD}}(\mathbf{x})$ outside this subset.

Hybrid Fraud-Aware Strategy: FRaUD++ Hybrid

The FRaUD++ Hybrid strategy (`HybridFraudPPQBCSampler`, reported as `fraudpp_hybrid`) combines the normalized FRaUD++ score (from Eq. 2) with normalized committee disagreement *(QBC)*. This balances the fraud-aware exploitation of FRaUD++ with model-based uncertainty exploration from a committee:

$$q_{\text{hybrid}}(\mathbf{x}) = \widetilde{q}_{\text{FRaUD++}}(\mathbf{x}) + w_{\text{QBC}}\, \widetilde{\text{QBC}}(\mathbf{x}), \tag{3}$$

where $\widetilde{q}_{\text{FRaUD++}}(\mathbf{x})$ denotes a min–max normalized version of $q_{\text{FRaUD++}}(\mathbf{x})$ and $\widetilde{\text{QBC}}(\mathbf{x})$ is the normalized disagreement score from the committee. The weight w_{QBC} is a fixed mixing hyperparameter that controls the relative strength of the QBC component. This hybrid score is then passed to the same diversity-aware batch selection step as FRaUD++, which helps reduce repeated sampling of near-duplicate high-probability points and encourages exploration of regions where the committee strongly disagrees.

Graph-Based Hybrid Strategy

Our second major contribution is **GraphHybrid** (`GraphFraudHybridSampler`, reported as `graph_hybrid`), a structure-aware strategy that uses local geometry among high-risk samples.

Step 1: High-risk candidate selection. In each iteration we select a candidate set $\mathcal{S}$ consisting of the top-m high-risk transactions according to the FRaUD++ score $q_{\text{FRaUD++}}(\mathbf{x})$ (or equivalently, the closely related fraud-aware score $s_0(\mathbf{x})$ used in the implementation). The value of m scales with the batch size and a configurable top-k factor, ensuring that graph construction focuses on a tractable but informative subset of the pool.

Step 2: Build k-NN graph. Using Euclidean distance in the standardized feature space (after z-score normalization of the features), we construct a k-nearest-neighbor graph

$$G = (\mathcal{S}, E), \qquad \mathcal{S} = \{\mathbf{x}_1, \ldots, \mathbf{x}_m\},$$

where each node is connected to its k closest neighbors. Graph construction is restricted to $\mathcal{S}$ to keep computation lightweight.

Step 3: Graph-based signals. On this graph we compute two local structural signals:

- **Hub score** $h(\mathbf{x})$: a density proxy defined as the inverse mean distance to neighbors,

$$h(\mathbf{x}) = \frac{1}{\frac{1}{|N(\mathbf{x})|}\sum_{\mathbf{z}\in N(\mathbf{x})} \text{dist}(\mathbf{x},\mathbf{z}) + \varepsilon},$$

  so that points in dense high-risk clusters obtain larger hub scores.
- **Bridge score** $b(\mathbf{x})$: a boundary-diversity measure computed from the fraud probabilities of the neighbors (and the point itself),

$$b(\mathbf{x}) = 0.7\,\text{Var}\big(\{p(\mathbf{z}) : \mathbf{z} \in N(\mathbf{x}) \cup \{\mathbf{x}\}\}\big) + 0.3\,\text{Range}\big(\{p(\mathbf{z}) : \mathbf{z} \in N(\mathbf{x}) \cup \{\mathbf{x}\}\}\big).$$

  where $\text{Range}(S) = \max S - \min S$. Larger $b(\mathbf{x})$ highlights points that lie near heterogeneous, high-contrast boundaries between fraud and non-fraud regions.

Both scores are min–max normalized across the candidate set $\mathcal{S}$ to produce $\widetilde{h}(\mathbf{x})$ and $\widetilde{b}(\mathbf{x})$ in $[0,1]$.

Step 4: Graph-Enhanced Score. A graph-enhanced fraud score is created by additively combining these bonuses with the base FRaUD++ score (Eq. 2):

$$q_{\text{graph}}(\mathbf{x}) = q_{\text{FRaUD++}}(\mathbf{x}) + w_{\text{hub}}\,\widetilde{h}(\mathbf{x}) + w_{\text{bridge}}\,\widetilde{b}(\mathbf{x}), \tag{4}$$

where w_{hub} and w_{bridge} are weights on the hub and bridge bonuses. In the implementation, these graph bonuses are applied only to the top-m candidates in $\mathcal{S}$; points outside $\mathcal{S}$ retain their original FRaUD++ scores.

Step 5: Final Hybrid Mix. The graph-enhanced score is then min–max normalized and mixed with normalized **QBC** disagreement:

$$q_{\text{graph-hybrid}}(\mathbf{x}) = \widetilde{q}_{\text{graph}}(\mathbf{x}) + w_{\text{QBC}}\,\widetilde{\text{QBC}}(\mathbf{x}), \tag{5}$$

where both $\widetilde{q}_{\text{graph}}$ and $\widetilde{\text{QBC}}$ are obtained by min–max normalization over the unlabeled pool, and w_{QBC} is a fixed mixing hyperparameter controlling the relative strength of the QBC component.

Step 6: Diversity-based batch selection. For both FRaUD++ Hybrid and GraphHybrid, the final batch is selected using the same two-stage, diversity-aware procedure: an exploitation stage that picks high-probability fraud candidates and an exploration stage that selects points with high hybrid scores, with both stages using farthest-point diversity in a PCA embedding as implemented in `utils.py`. This reduces redundant label requests and improves coverage across distinct fraud subregions.

Why Hybrid Strategies Work

The hybrid approaches address the three known failure modes of AL in imbalanced settings:

1. **Uncertainty collapse** Uncertainty sampling often selects majority-class outliers; fraud priors counterbalance this.
2. **Mode collapse in exploitation** Pure fraud-prior sampling repeatedly queries near-identical examples; diversity and hybrid committee disagreement (QBC) prevent this.
3. **Structural blind spots** GraphHybrid uses local geometry to identify fraud hubs and potential new fraud modes.

Together, these properties yield dramatically better label efficiency in our experiments.

Algorithmic Summary

A complete pseudocode description of the AL loop and hybrid acquisition functions is provided in Appendix A.

Runtime Considerations

Table 1: Average runtime per active learning iteration across strategies.

strategy	eval_time_mean	eval_time_std	max_labels
cost_balanced	0.168000	0.078000	5000
entropy	0.131000	0.081000	5000
fraud	0.158000	0.094000	5000
fraudpp	0.239000	0.590000	5000
fraudpp_hybrid	0.146000	0.092000	5000
graph_hybrid	0.135000	0.083000	5000
margin	0.113000	0.064000	5000
qbc	0.137000	0.096000	5000
random	0.148000	0.081000	5000

Runtime per AL iteration is summarized in Table 1. In our measurements, all strategies have comparable per-iteration cost, with `graph_hybrid` showing runtime similar to `entropy` and `qbc`, and slightly faster than `random`. The `fraudpp` strategy is the slowest on average due to its additional rarity computations. For strict time budgets, `cost_balanced`, `fraud`, and the hybrid strategies (`fraudpp_hybrid`, `graph_hybrid`) offer a good speed–accuracy trade-off while still substantially outperforming `random`, `entropy`, `margin`, and `qbc` in label efficiency.

Experimental Setup

Dataset

We use the public Kaggle Credit Card Fraud Detection dataset (Kaggle, 2018), which contains 284,807 transactions with 492 frauds ($\approx 0.17\%$ positives). All features are anonymized continuous variables (V1–V28) along with `Time`, `Amount`, and a binary fraud label. Following standard practice in pool-based AL, we perform a stratified train/test split: a fixed hold-out test set is never labeled and is used only for evaluation, while the AL loop repeatedly trains on the growing labeled subset of the training pool.

A small subset of the training data is additionally held out as a validation set for LightGBM early stopping (when enabled in the configuration). The operating threshold corresponding to 0.1% FPR is computed directly from the held-out test set by applying a threshold-selection routine to the model's probability scores on that set.

Preprocessing

We follow standard preprocessing for the Kaggle credit card dataset. The `Time` column is removed because it carries no discriminative information after anonymization, and the PCA-derived features V1–V28 are used as provided. The `Amount` feature is standardized using z-score normalization with a `StandardScaler` fit on the training split.

For the graph-based strategy (**graph_hybrid**), at each AL iteration we construct a k-nearest-neighbor graph ($k = 10$) on a candidate subset consisting of the most suspicious transactions according to the fraud-aware FRaUD++ score used in our implementation. Distances are computed in this preprocessed feature space using Euclidean distance. Graph construction is repeated each round only on this small candidate subset, keeping computation lightweight.

Models and Baselines

The base classifier in all experiments is **LightGBM** (Ke et al., 2017) with class weighting enabled. LightGBM is used because it trains efficiently on continually growing labeled sets and captures non-linear interactions common in fraud data. Our framework supports optional Platt-style probability calibration via `CalibratedClassifierCV`, but in the experiments reported here we set `use_calibration = false` and compute all metrics (including threshold-based measures) using LightGBM's native probability outputs.

All strategies operate on the *same* model, labeled seed set, and batch size. The evaluated acquisition functions are:

- **random**: uniform sampling;
- **entropy**: Shannon entropy of the model's predicted fraud probabilities;
- **margin**: distance to the decision boundary;
- **QBC**: using a committee of heterogeneous models (Logistic Regression, Random Forest, and an MLP);
- **cost_balanced**: entropy weighted by inverse class frequency;
- **fraud**: fraud-prior scoring using the full FRaUD base score in the implementation;
- **fraudpp**: fraud-aware FRaUD++ score combining uncertainty, focal prior, boundary proximity, and rarity;
- **fraudpp_hybrid**: FRaUD++ combined with QBC disagreement;

- **graph_hybrid**: graph-augmented FRaUD++ combined with QBC disagreement.

This list matches the exact strategies implemented in `strategies.py` and reported in our CSV summaries.

Active Learning Loop Settings

Each run begins with a stratified initial labeled seed of size $s_{\text{seed}} = 1400$. At every iteration the active learner:

1. trains a LightGBM classifier on the current labeled set,
2. scores the unlabeled pool with the acquisition function,
3. selects a batch of $b_t = 400$ samples from the pool,
4. obtains their labels,
5. updates the labeled set and repeats.

Experiments are run for labeling budgets between 1400 and 5000 labeled transactions, in increments of 400 samples per round, starting from the initial seed of 1400 labeled examples. All results are averaged over **three random seeds** (0, 1, 2), as reflected in the aggregated CSV summaries.

Implementation Details

All experiments were executed on a standard workstation (CPU-based LightGBM; no GPU is required). Hyperparameters (such as learning rate, number of leaves, and depth) were tuned once in preliminary runs and then held fixed across all strategies to ensure fair comparison. PCA dimensionality reduction is used only inside the farthest-point diversity helper for hybrid methods, where a compact embedding can optionally be computed before selecting a diverse batch.

Our implementation relies on standard Python libraries: `numpy` and `pandas` for data manipulation, `scikit-learn` for preprocessing, validation, and the QBC ensemble, `lightgbm` for the base classifier, and `matplotlib` for plotting.

Evaluation Metrics

We report several metrics relevant to extreme imbalance and financial risk:

- **AUPRC**: primary metric for imbalanced classification;
- **AUROC**: complementary global ranking measure;
- **Recall@0.1% FPR**: domain-relevant operating point using a threshold chosen to achieve approximately 0.1% FPR on the held-out test set;
- **Frauds Found**: cumulative number of frauds discovered as labeling progresses;
- **Learning Curves**: performance as a function of labeling budget;
- **Profit**: expected value using asymmetric fraud/false-alarm costs.

No plots appear in this section; all visualizations are deferred to Section .

Experiments and Evaluation

In this section we evaluate all nine AL strategies along several axes: learning dynamics across labeling budgets, final performance at 5000 labels, early-stage label efficiency, sensitivity to analyst capacity, robustness across seeds, and runtime. All curves are averaged over three random seeds as described in Section . Unless otherwise noted, ranking-based metrics (AUPRC, AUROC, recall@0.1% FPR, and profit) are computed on the held-out test set using the ground-truth labels, and `frauds_found` denotes the cumulative number of frauds recovered in the labeled portion of the training pool up to a given budget.

Learning Curves Across Budgets

We first examine how performance evolves as more transactions are labeled.

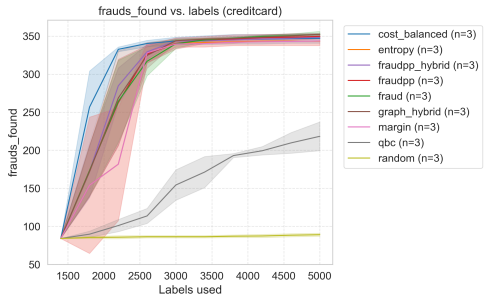

Figure 1: Cumulative frauds found as a function of labeled transactions (average over 3 seeds). Fraud-aware methods (`fraud`, `fraudpp`, `fraudpp_hybrid`, `graph_hybrid`) and `cost_balanced` recover substantially more frauds than random, entropy, margin, and QBC across the entire budget range.

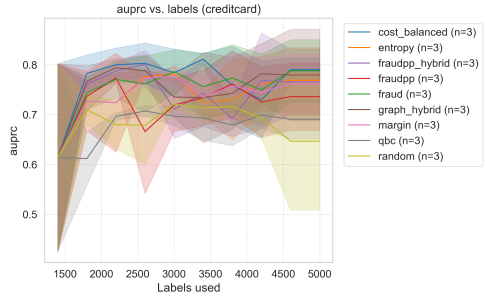

Figure 2: AUPRC vs. labeling budget. Fraud-aware and hybrid methods reach high AUPRC regions much earlier than random, entropy, margin, and QBC, indicating better label efficiency.

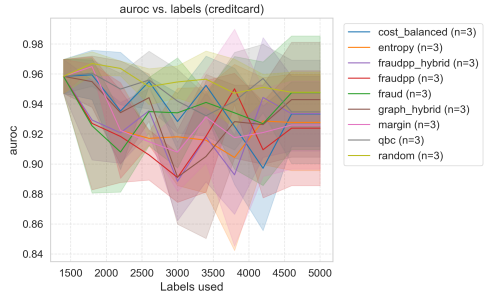

Figure 3: AUROC vs. labeling budget. All methods improve with more labels and reach similarly high AUROC values at larger budgets. Differences between strategies in AUROC are smaller than in AUPRC or `frauds_found`, so we focus primarily on precision–recall behavior and fraud discovery when comparing methods under extreme imbalance.

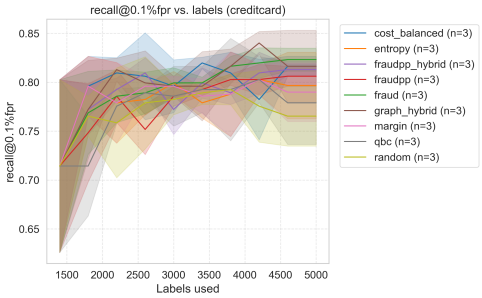

Figure 4: Recall at 0.1% FPR over budgets. Fraud-aware and hybrid strategies maintain higher recall in the low-FPR regime that is realistic for financial review settings.

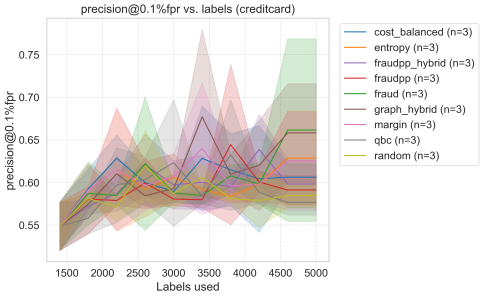

Figure 5: Precision at 0.1% FPR. Precision for `entropy`, `margin`, and the hybrid methods improves over the budget range and exhibits moderate variability across seeds, while `qbc`, `random`, and `cost_balanced` show relatively tighter spreads. Overall, the hybrid methods achieve competitive precision at the low-FPR operating point.

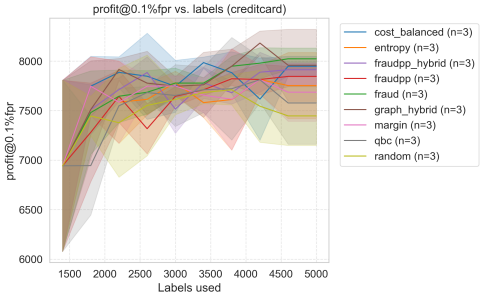

Figure 6: Profit-style metric at 0.1% FPR (using asymmetric fraud/false-alarm costs). Fraud-aware and hybrid strategies consistently yield higher expected utility than simple uncertainty baselines.

Summary. Across all learning curves, a consistent picture emerges: random and margin require many more labeled transactions to reach a practically useful region. Entropy and QBC improve faster but still lag behind the fraud-aware strategies. `fraud`, `fraudpp`, `fraudpp_hybrid`, `graph_hybrid`, and `cost_balanced` reach strong AUPRC, AUROC, recall, and profit levels at much lower budgets, reflecting substantially better label efficiency.

Final Performance at 5000 Labels

We now compare all strategies at the final labeling budget of 5000 transactions, where each method has seen the same number of labels.

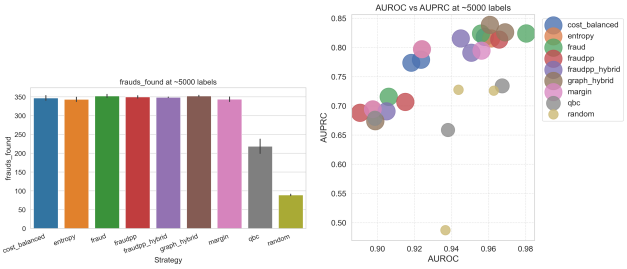

Figure 7: Left: cumulative frauds found at 5000 labels. Right: AUROC vs. AUPRC at 5000 labels. Fraud-aware baselines (`fraud`, `fraudpp`, `cost_balanced`) and hybrids (`fraudpp_hybrid`, `graph_hybrid`) form the top-performing group, with `entropy` and `margin` slightly behind but still substantially stronger than `random` and `qbc`, which are the weakest methods.

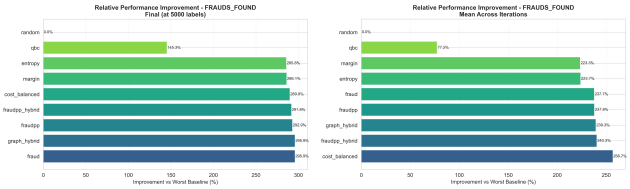

Figure 8: Relative improvement in cumulative frauds found over the worst-performing method (random). Left: final budget (5000 labels). Right: average over all budgets. The best methods detect up to $\sim$2.96$\times$ more frauds than random sampling.

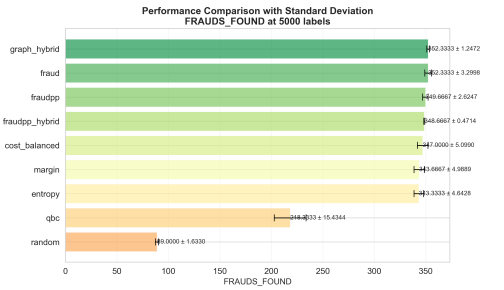

Figure 9: Mean $\pm$ standard deviation of `frauds_found` at 5000 labels across seeds. Fraud-aware and hybrid methods achieve the highest means with relatively tight error bars; QBC and entropy show both lower means and higher variance.

At 5000 labels, the fraud-aware and hybrid strategies clearly dominate `random` and `qbc` in terms of `frauds_found`, and also outperform `entropy` and `margin` in cumulative fraud discovery. The fraud-aware baselines (`fraud`, `fraudpp`, `cost_balanced`) and the hybrids (`fraudpp_hybrid`, `graph_hybrid`) occupy the upper part of the AUPRC plane, with AUPRC values around 0.74–0.79. AUROC values are high and relatively similar for almost

all methods (approximately 0.93–0.95), reflecting the limited discriminative power of AUROC under extreme class imbalance; accordingly, we place more emphasis on AUPRC and `frauds_found` when comparing strategies. The improvement plots confirm that the best methods detect up to $\sim 2.96\times$ more frauds than `random` at the final budget (about 296% improvement in `frauds_found`).

Early-Stage Label Efficiency

Real fraud review teams often cannot wait until the full budget of 5000 labels to see gains, so we also assess how quickly each strategy delivers useful performance. Instead of fixing the labeling budget and comparing metrics, we invert the perspective and ask: *how many labeled examples does each method need to reach a given performance level?*

In this section we focus on a relatively strict target of AUPRC 0.75, which corresponds to a practically useful detection quality under extreme imbalance. Table 2 reports, for each strategy that attains this target, the average number of labels required across seeds.

Table 2: Labels needed to reach AUPRC 0.75 (lower is better).

strategy	labels needed mean	labels needed std	n seeds	relative gain (%)
cost_balanced	1933.33	230.94	3.00	0.00
fraudpp_hybrid	1933.33	230.94	3.00	0.00
graph_hybrid	1933.33	230.94	3.00	0.00
fraud	2066.67	230.94	3.00	−6.90
fraudpp	2066.67	230.94	3.00	−6.90
entropy	2333.33	461.88	3.00	−20.69
margin	2333.33	461.88	3.00	−20.69

*T*able 2 shows that, among the methods that reach AUPRC 0.75 within the evaluated label budgets, the most label-efficient are `cost_balanced`, `fraudpp_hybrid`, and `graph_hybrid`, which attain this target with the fewest labels (about 1933 on average). `fraud` and `fraudpp` require slightly more labels (roughly 7% more), while `entropy` and `margin` need about 20% more labels than the best group. Simpler baselines such as `random` and `qbc` do not appear in the table because they never reach AUPRC 0.75 within the considered budgets.

Robustness and Variance Across Seeds

Finally, we analyze robustness across random seeds.

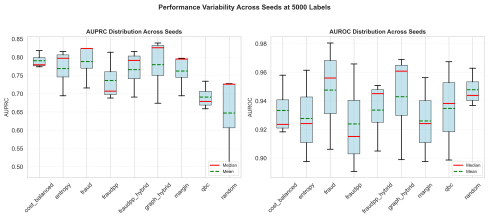

Figure 10: Distribution of `frauds_found` across random seeds at 5000 labels. Fraud-aware and hybrid strategies show tighter spreads than entropy and QBC, which are more sensitive to seed choice.

Combined with the error bars in Figure 9, the boxplot indicates that the proposed hybrids not only perform well on average but also exhibit stable behavior across different initializations, an important property in operational fraud detection systems. (Complete configuration files and logs are available in the GitHub repository.)

Conclusion and Future Work

In this work, we systematically evaluated a family of fraud-oriented AL strategies for highly imbalanced credit card transaction data. Using a unified pipeline and controlled experiments across multiple labeling budgets and random seeds, we found that both `fraudpp_hybrid` and `graph_hybrid` substantially improve label efficiency compared to traditional uncertainty- and committee-based methods. At a budget of 5000 labels, hybrid strategies detect between 61% and 296% more frauds than weaker baselines such as `qbc` and `random`, while maintaining competitive median AUPRC 0.826 . Moreover, they exhibit lower variance across seeds and greater robustness under constrained analyst capacity. These findings highlight the importance of incorporating fraud-aware priors, calibrated uncertainty, and graph-structural cues into the acquisition loop.

Beyond their empirical performance, the proposed hybrids demonstrate a practical advantage for real-world fraud review teams: they discover meaningful fraud clusters early (1500–3000 labels) and remain stable across both budgets and committee disagreement noise. This suggests that hybrid scoring functions—particularly those blending uncertainty, class priors, and local manifold structure—provide a promising direction for rare-event detection in cost-sensitive domains.

Future work. Several extensions can further advance this line of research. First, integrating deep or learned representations (e.g., transformers or graph neural networks) may provide richer structural information than PCA and *k*NN graphs. Second, adaptive committee construction or dynamic weighting between FRaUD++ and QBC disagreement could improve stability in later rounds. Third, extending the framework to streaming or concept-drift settings is essential for production fraud systems, where attack patterns evolve rapidly.

Appendix

A. Implementation Details

This appendix provides technical details needed for full reproducibility while keeping the main report clear and concise.

Libraries. All experiments were implemented in Python using:

- `LightGBM` for the base classifier,
- `scikit-learn` for preprocessing, calibration, and the QBC ensemble,
- `numpy`, `pandas` for data manipulation,
- `faiss` (optional) or `sklearn.neighbors` for *k*NN graph construction,
- `matplotlib` and `seaborn` for plotting.

B. Preprocessing Details

Feature Scaling. All features were standardized using z-score normalization:

$$\tilde{x}_j = \frac{x_j - \mu_j}{\sigma_j},$$

where μ_j and σ_j are computed from the initial labeled seed $\mathcal{L}_0$.

Graph Construction. For `graph_hybrid`, we construct a kNN graph at each AL iteration:

- Candidate set $\mathcal{S}$: top $m = 500$ transactions ranked by $p_\theta(y{=}1|x)$.
- Metric: Euclidean distance in standardized feature space.
- Neighbors: $k = 10$.

Graph-based centrality and bridge scores are then min–max normalized.

C. Hyperparameter Configuration

Table 3 lists the base model settings used for all experiments.

Table 3: LightGBM hyperparameters.

Parameter	Value
num_leaves	31
learning_rate	0.05
n_estimators	300
class_weight	`balanced`
min_child_samples	20
subsample	0.8
colsample_bytree	0.8

These parameters were tuned once on the validation set and held fixed for all AL strategies to ensure fairness.

Calibration uses Platt scaling via `CalibratedClassifierCV(method="sigmoid")`.

D. Active Learning Loop Pseudocode

Algorithm 1 Pool-Based Active Learning Loop

Initialize labeled set $\mathcal{L}_0$, unlabeled pool $\mathcal{U}_0$
for iteration $t = 1, \ldots, T$ **do**
  Train LightGBM model f_{θ_t} on $\mathcal{L}_{t-1}$
  Compute calibrated scores $p_{\theta_t}(y{=}1 \mid x)$
  Compute acquisition scores $q_t(x)$ for all $x \in \mathcal{U}_{t-1}$
  Select top m candidates by $q_t(x)$
  Apply PCA + farthest-point sampling to select batch B_t
  Query labels for B_t and update:

$$\mathcal{L}_t \leftarrow \mathcal{L}_{t-1} \cup B_t, \quad \mathcal{U}_t \leftarrow \mathcal{U}_{t-1} \setminus B_t$$

end for

E. Analyst Capacity Sensitivity

In practice, the number of cases an analyst team can review per round (capacity K) is a real operational constraint. We therefore evaluate how sensitive each strategy is to changes in K.

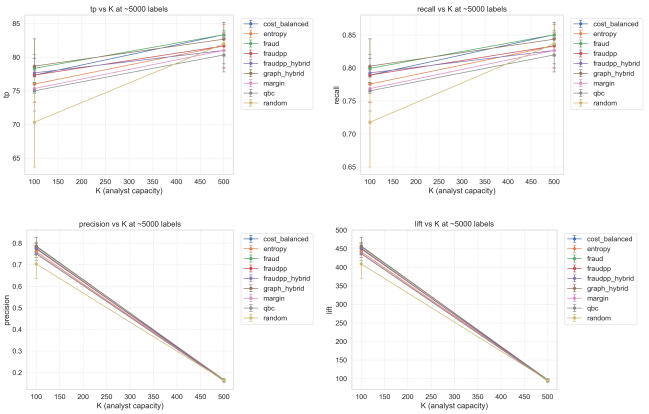

Figure 11: Capacity sensitivity at 5000 labels (late stage). Increasing analyst capacity K improves performance for all methods: true positives and recall increase with K, while precision and lift decrease as more cases are reviewed. Fraud-aware and hybrid strategies, together with the `cost_balanced` baseline, generally achieve higher recall, precision, and lift than `random` and `qbc` across the tested capacities.

Takeaway. At this late stage of the active learning process, increasing analyst capacity K improves true positives and recall for all strategies, while precision and lift naturally decrease as more transactions are inspected. Fraud-aware and hybrid methods, along with `cost_balanced`, tend to deliver better performance than `random` and `qbc` at a given capacity, which is especially important when analyst resources are limited.

References

Ash, J. T., Zhang, C., Krishnamurthy, A., Langford, J., & Agarwal, A. (2020). Deep batch active learning by diverse, uncertain gradient lower bounds. In *International conference on learning representations (iclr)*.

Dal Pozzolo, A., Caelen, O., Le Borgne, Y.-A., Waterschoot, S., & Bontempi, G. (2015). Calibrating probability with undersampling for unbalanced classification. *2015 IEEE Symposium Series on Computational Intelligence*, 159–166. doi: 10.1109/SSCI.2015.33

Kaggle. (2018). *Credit card fraud detection dataset.* https://www.kaggle.com/mlg-ulb/creditcardfraud. (Accessed: 2025-11-03)

Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., … Liu, T.-Y. (2017). Lightgbm: A highly efficient gradient boosting decision tree. In *Advances in neural information processing systems (neurips)*.

Lewis, D. D., & Gale, W. A. (1994). A sequential algorithm for training text classifiers. In *Proceedings of the 17th annual international acm sigir conference on research and development in information retrieval* (pp. 3–12).

Sener, O., & Savarese, S. (2018). Active learning for convolutional neural networks: A core-set approach. In

International conference on learning representations (iclr).

Seung, H. S., Opper, M., & Sompolinsky, H. (1992). Query by committee. In *Proceedings of the fifth annual workshop on computational learning theory (colt)* (pp. 287–294).

Wu, Z., Pan, S., Chen, F., Long, G., Zhang, C., & Yu, P. S. (2020). A comprehensive survey on graph neural networks. *IEEE Transactions on Neural Networks and Learning Systems*, *32*(1), 4–24.