Active Learning for Power Grid Security Assessment: Reducing Simulation Cost with Informative Sampling

Gašper Leskovec
Jožef Stefan Institute
Slovenia
leskovecg@gmail.com

Costas Mylonas
UBITECH
Greece
kmylonas@ubitech.eu

Klemen Kenda
Jožef Stefan Institute
Slovenia
klemen.kenda@ijs.si

Abstract

Power grid security assessment under the N-1 criterion requires extensive contingency simulations, which are computationally intensive and costly to label. In this work, we explore the use of active learning (AL) to train binary classifiers that can accurately predict the outcome of contingency scenarios using fewer labeled samples. We evaluate several AL strategies, such as entropy, margin, and uncertainty sampling against a random baseline. Our results show that AL methods achieve the same predictive performance with significantly fewer labels, reducing labeling effort and simulator runtime. These findings demonstrate the effectiveness of integrating AL with power system simulators to enable scalable and efficient N-1 security assessment without sacrificing model accuracy.

Keywords

active learning, smart grids, security assessment, simulation cost reduction

1 Introduction

Ensuring secure operation of power systems under the N-1 criterion is a cornerstone of grid reliability. The criterion requires that the system remains within operational limits following the loss of any single component (e.g., line, transformer, or generator). In practice, this involves simulating a large number of contingencies and checking for violations of thermal or voltage constraints. While essential, such simulations are computationally intensive, particularly when performed on high-fidelity grid models, and their interpretation often requires expert judgment. This creates a bottleneck for both real-time applications and large-scale scenario analyses, where scalability and efficiency are important.

Classical approaches to N-1 assessment rely on exhaustive AC power flow simulations combined with contingency ranking heuristics such as performance indices (PIs). While useful for screening, these heuristics may mis-rank contingencies or overlook borderline cases due to masking effects [3]. Moreover, exhaustive analysis does not scale well with system size, making it unsuitable for fast or repeated assessments.

To overcome these challenges, researchers have proposed machine learning (ML) and deep learning (DL) approaches that approximate N-1 contingency outcomes directly from operating point features. One of the earliest contributions in this direction applied convolutional neural networks (CNNs) to contingency datasets, showing that deep models could achieve over 99% accuracy in detecting insecure cases while being more than 200 times faster than traditional power flow calculations [1]. Building on this, more recent work explored pooling-ensemble multi-graph learning to design scalable contingency screening schemes based on steady-state information, demonstrating improved adaptability for large-scale systems [2]. These approaches enable fast security screening without solving power flows for every contingency. However, their reliability hinges on the availability of large labeled datasets covering all relevant operating points and contingencies. Such datasets are typically generated by running exhaustive offline N-1 simulations, which is computationally expensive, or require significant expert effort to label secure versus insecure cases. This dependence on costly and large-scale data generation remains a major limitation of existing ML-based frameworks for steady-state security assessment.

To reduce labeling costs, AL has recently been explored in other areas of power systems. For example, authors of [5] used AL to enhance stability assessment and dominant instability mode identification, showing that models could be trained with far fewer labeled samples while maintaining accuracy. Similarly, authors of [4] demonstrated an AL-enhanced digital twin for day-ahead load forecasting, where the model iteratively refined predictions by querying only the most uncertain cases. These studies confirm the potential of AL to reduce expert effort and simulation cost by strategically selecting informative samples. However, AL has not yet been applied to N-1 steady-state security assessment, where the need to cut down on contingency simulations is especially critical.

In this work, we propose a novel framework for AL driven N-1 security assessment. Our contributions are threefold:

(1) We design a binary classification model that predicts whether a given contingency is secure or insecure based on steady-state features.
(2) We integrate AL strategies (entropy, margin, and uncertainty sampling) with the classifier to selectively query the most informative contingencies for simulation, reducing the number of labels required.
(3) We demonstrate through a case study that our approach achieves the same predictive accuracy as fully supervised baselines while reducing simulation cost and labeling effort by up to 40–50%.

This work provides the first evidence that AL can be directly leveraged for N-1 security assessment, offering a scalable and label-efficient alternative to exhaustive simulation or purely supervised ML approaches.

2 Methodology

We study whether pool-based AL can reduce the number of expensive N-1 simulations ("labels") while keeping prediction quality for binary *secure vs. insecure* classification.

SiKDD 2025, Ljubljana, Slovenia

ACM ISBN 978-x-xxxx-xxxx-x/YYYY/MM
https://doi.org/10.70314/is.2025.sikdd.11

Table 1: Dataset and system description (digital twin of the Greek transmission network).

Attribute	Value
Test system	35 buses, 46 lines, 135 generators, 110 static generators, 20 loads
Power flow solver	AC load flow (Newton–Raphson), via `pandapower`
Contingencies (N–1)	Line outages (all lines except idx 45), generator outages (all)
Total contingency cases	8 769
Secure / Insecure	51.28% / 48.72%
Feature dimensionality	271 features total
Feature groups	`load_`: 20, `gen_`: 135, `sgen_`: 110

2.1 Data and labels from a digital twin

We use a steady-state digital twin of the transmission grid. For each timestamp we solve the base-case AC power flow, then apply the N-1 criterion by removing each line/transformer/generator in turn and re-solving. An operating point is labeled *secure* if the base case and all contingencies satisfy limits (bus voltages $\in [0.90, 1.10]$ p.u., line loading $\leq 100\%$); otherwise it is *insecure*. Non-convergent power flows are labeled *insecure*.

The test system is a digital twin derived from the topology of the Greek transmission network (35 buses, 46 lines, 135 generators, 110 static generators, 20 loads). AC load flows are computed with the Newton–Raphson method in `pandapower`. N-1 contingencies include all line outages (excluding line index 45) and all generator outages. Table 1 summarizes the dataset.

2.2 Time-aware train/validation/test split

Samples are sorted by timestamp. The AL *training/pool* comes from earlier windows, while the *test* set is the most recent slice and is never used for training or querying. This avoids temporal leakage and mimics deployment where we predict on future data. A small validation split is carved from the training era for early checks.

2.3 Classifier and hyperparameters

Our base model is a Random Forest (RF) because it is fast, robust and provides class-probability posteriors needed by uncertainty-based AL. Across runs we vary hyperparameters in realistic ranges: $n_{\text{estimators}} \in [200, 1500]$, `max_depth` $\in \{18, 20, 24, 25, 28, 30, 35, 40, \text{None}\}$, `min_samples_split` $\in \{2, 4\}$, `min_samples_leaf` $\in \{1, 2, 3\}$, `class_weight` $\in \{\texttt{balanced}, \texttt{balanced\_subsample}\}$. We use seeds $\{42, 1337\}$ for reproducibility.

Classifier dependence. We use Random Forests for probability outputs and fast retraining inside the AL loop. While AL's relative gains often transfer across probabilistic classifiers, we did not perform a systematic model sweep here. Evaluating logistic regression and gradient-boosted trees under the same AL protocol is left to future work.

2.4 Pool-based AL loop

We follow the standard pool-based AL recipe:

(1) Start with an initial labeled set of size i and an unlabeled pool.
(2) Train the RF on the current labeled set; score the pool to obtain class-probability vectors $p(x)$.
(3) Select the next batch of b samples using one of the query strategies below.
(4) Query the simulator for labels of the chosen batch (expensive step); add them to the labeled set.
(5) Repeat for a fixed number of iterations or until the budget is exhausted.

We sweep budgets across runs: $i \in \{100, \ldots, 500\}$, $b \in \{50, \ldots, 200\}$, and up to 40 iterations, which lets us trace long learning curves.

Query strategies. We compare: (i) **Random** (baseline); (ii) **Least-confident** (*uncertainty*): score $1 - \max_c p_c(x)$; (iii) **Margin**: negative gap between top-2 probabilities; (iv) **Entropy**: $-\sum_c p_c(x) \log p_c(x)$. All three uncertainty policies operate on the same RF posteriors and therefore often rank samples similarly.

2.5 Evaluation

After each iteration we evaluate on the fixed test set. At each AL round we retrain the RF from scratch on the enlarged labeled set; new labels are added to training only; the pool remains unlabeled. For each strategy we run multiple configurations and both seeds, then align results by total labeled samples and average across runs to obtain strategy-level learning curves. Unless noted otherwise, TTT values in the main figures are computed on these averaged curves. Appendix A.1 (Table 4a) reports per-run TTT (mean ± std), which is larger due to variability across initial sizes i, batch sizes b, and seeds.

2.6 Metrics

We report Accuracy and ROC AUC on the test set, plus two label-efficiency metrics: **Time-to-Target** (TTT), the smallest number of labeled samples needed for the *average* curve of a strategy to reach a target (e.g., ACC≥0.92 or AUC≥0.98); and **AULC** (Area Under the Learning Curve), computed by trapezoidal integration of metric vs. total labeled. Because simulator seconds per call are roughly constant, relative cost/time savings are well approximated by label savings derived from TTT.

Additional classification metrics. Besides Accuracy and ROC AUC we also track **Precision**, **Recall**, **F1** and the **False Negative Rate (FNR)** on the fixed test set at every AL round. Let TP, FP, FN, TN be counts on the test set. We use the standard definitions: Precision = TP/(TP + FP), Recall = TP/(TP + FN), F1 = $2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$, FNR = FN/(FN + TP) = 1 − Recall. We report *mean ± std across runs/seeds*, and we extract TTT-style thresholds for these metrics when relevant.

3 Results

Figure 1 and Figure 2 show learning curves (averaged across seeds). Across the budget range, all three uncertainty-based policies (entropy, margin, uncertainty) dominate the `random` baseline in both Accuracy and ROC AUC; the area under the learning curve (AULC) is consistently higher.

Table 3 summarizes KPIs used in the paper. At the most important targets, AL reaches the same performance with far fewer labels: at **ACC** ≥ 0.92, AL needs about **500** labels vs. **1 040** for `random` (∼ **52%** fewer); at **AUC** ≥ 0.98, AL needs **580** vs. **960** (∼ **40%** fewer). Final metrics at the maximum budget are also higher for AL (ACC 0.917±0.005 and AUC 0.983±0.002) than for

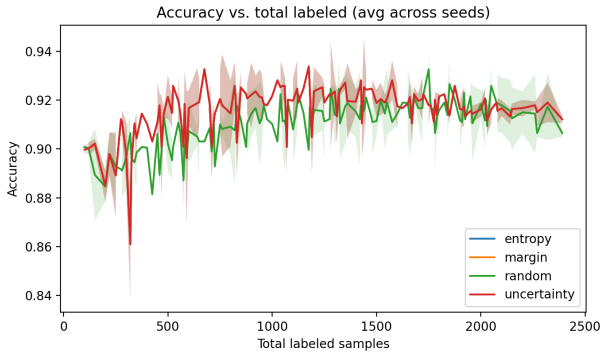

Figure 1: Accuracy vs. total labeled samples (mean ± std across runs). ***(Note: entropy, margin, and uncertainty overlap almost perfectly on this dataset—so the three AL curves/bands lie on top of each other; Random is shown separately for contrast)***

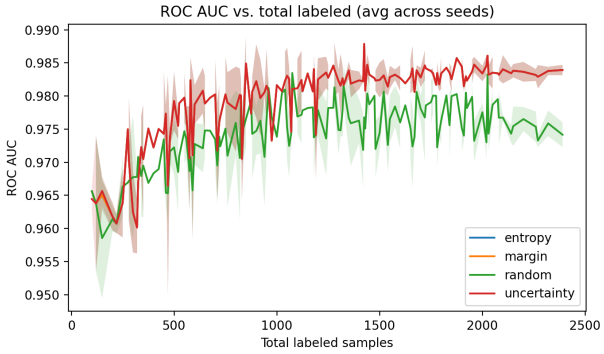

Figure 2: ROC AUC vs. total labeled samples (mean ± std across runs). ***(Note: entropy, margin, and uncertainty overlap almost perfectly on this dataset—so the three AL curves/bands lie on top of each other; Random is shown separately for contrast)***

Table 2: Final test metrics at maximum budget (mean ± std across runs).

Strategy	Accuracy	ROC AUC
entropy	0.917 ± 0.005	0.983 ± 0.002
margin	0.917 ± 0.005	0.983 ± 0.002
uncertainty	0.917 ± 0.006	0.983 ± 0.004
random	0.916 ± 0.010	0.977 ± 0.004

`random` (ACC 0.916±0.010 and AUC 0.977±0.004). Differences at the easier target **ACC** $\geq$ 0.90 are small (all reach it by ~100–120 labels), which is expected for a low threshold.

On high AUC values. The time-aware split still yields a separable test set for this case study (AUC $\approx$ 0.98). This likely reflects informative steady-state features and balanced classes, not overfitting to the test era. That said, harder, more imbalanced systems may reduce AUC and amplify AL gains; we treat this as a scope limitation.

Precision, Recall, F1 and FNR.. The additional metrics mirror the ACC/AUC trends: entropy, margin, and uncertainty produce higher AULC and reach target quality with fewer labels than `random`. At targets Precision/Recall/F1 $\geq$ 0.90 and FNR $\leq$ 0.10, the uncertainty-based policies consistently hit the thresholds earlier on the average curves, confirming that the AL gains are not specific to a single metric. Shaded bands (std across runs) show the same ordering stability observed for ACC/AUC. Full KPI values and TTT thresholds for P/R/F1/FNR are provided in Appendix A.2 (Table 4b).

Next, we compare label efficiency using Time-to-Target (TTT). Figures 3 and 4 show TTT for accuracy targets 0.90 and 0.92, while Figures 5 and 6 show TTT for AUC targets 0.97 and 0.98. At the **easy** target **ACC** $\geq$ 0.90 all strategies reach the goal after about **100–120** labels (uncertainty sometimes at 120 due to seed/batch noise). At the more demanding **ACC** $\geq$ 0.92 target, active-learning policies need about **500** labels, whereas `random` needs **1 040** (i.e., ~**52%** fewer labels). For **AUC** $\geq$ 0.97, AL reaches the target at **275** labels vs. **325** for `random` (~**15%** fewer), and for **AUC** $\geq$ 0.98 at **580** vs. **960** (~**40%** fewer). These reductions translate directly into lower simulation time when the average time per labeling call is roughly constant.

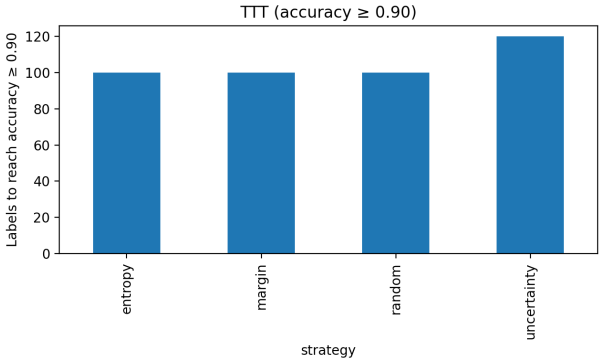

Figure 3: TTT (Accuracy $\geq$ 0.90): computed on the strategy-level average curve; per-run variability (mean ± std) is reported in Appendix.

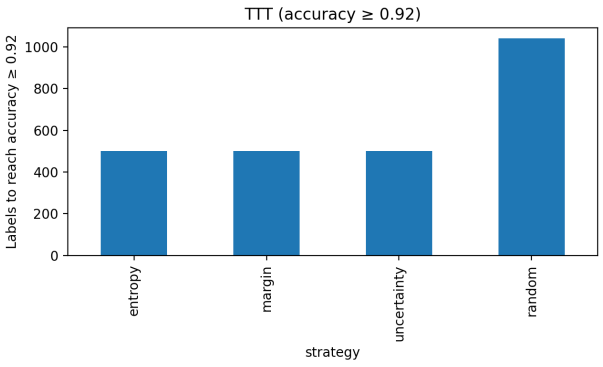

Figure 4: TTT (Accuracy $\geq$ 0.92): computed on the strategy-level average curve; per-run variability (mean ± std) is reported in Appendix.

Overall, uncertainty-based AL strategies consistently beat `random` at the harder targets (ACC 0.92 and AUC 0.98) while performing similarly at the easier ACC 0.90 threshold; final performance at the maximum budget remains high (ACC 0.917±0.005,

Table 3: KPIs by strategy (averaged across runs). Final metrics reflect per-run means; see Table 2 for mean±std.

	AULC		TTT (labels)				Final	
Strategy	acc	auc	acc ≥ 0.90	acc ≥ 0.92	AUC ≥ 0.97	AUC ≥ 0.98	ACC	AUC
entropy	0.92	0.98	100	500	275	580	0.917	0.983
margin	0.92	0.98	100	500	275	580	0.917	0.983
random	0.91	0.97	100	1 040	325	960	0.916	0.977
uncertainty	0.92	0.98	120	500	275	580	0.917	0.983

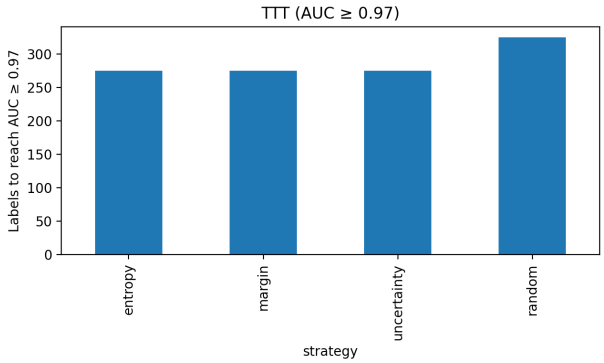

Figure 5: TTT (AUC ≥ 0.97): computed on the strategy-level average curve; per-run variability (mean ± std) is reported in Appendix.

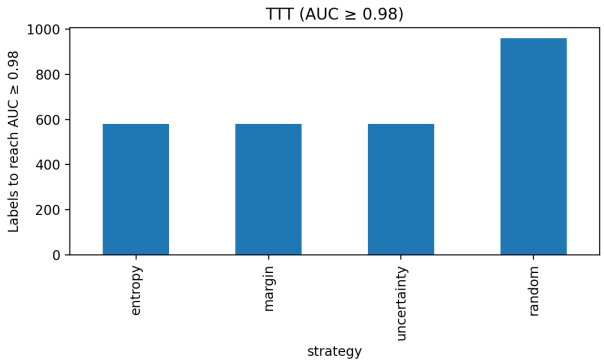

Figure 6: TTT (AUC ≥ 0.98): computed on the strategy-level average curve; per-run variability (mean ± std) is reported in Appendix.

AUC 0.983±0.002 for AL vs. ACC 0.916±0.010, AUC 0.977±0.004 for `random`).

4 Conclusion

This paper demonstrates that AL is a viable strategy for reducing simulation costs in power-grid security assessment. By selectively querying informative contingencies, we cut labels (and thus simulator calls) by **about 52% at ACC** ≥ 0.92 (500 vs. 1 040 with random) and **about 40% at AUC** ≥ 0.98 (580 vs. 960), without sacrificing final performance (AL: ACC 0.917±0.005, AUC 0.983±0.002; random: ACC 0.916±0.010, AUC 0.977±0.004; see Table 2). Fewer simulator calls translate into shorter training times and lower computational and memory requirements, which are particularly important for real-time or resource-constrained applications. Moreover, integrating AL within a digital-twin pipeline enables a feedback loop in which the classifier continuously refines itself using only the most informative contingencies. These findings suggest that exhaustive N-1 simulations are not always necessary for reliable security assessment, paving the way for more scalable and efficient grid-analysis tools.

The present study focuses on a single test system and a Random Forest classifier. In future work we plan to evaluate the proposed framework on larger and more diverse grid topologies (e.g., IEEE 39-bus, 118-bus or national transmission networks) and under varying operating conditions. Another direction is to explore more advanced models such as gradient-boosting machines, deep neural networks or graph neural networks, which may capture complex relationships among grid variables. We also intend to investigate alternative sampling strategies—including diversity-based selection, query-by-committee and Bayesian AL to further improve label efficiency. Finally, extending the methodology to multi-contingency (N-k) and dynamic security assessments (e.g., transient stability) will broaden its applicability in future smart-grid deployments.

Reproducibility

Code, analysis scripts, and a dataset to reproduce all figures and tables will be released at https://github.com/HumAIne-JSI/smart-energy-ea.

Acknowledgements

This work was supported by European Union's funded Project HUMAINE [grant number 101120218]. The authors acknowledge the use of LLMs for language optimization. While the LLMs contributed to enhancing efficiency and refining the presentation of this work, all conceptual frameworks, analyses, and interpretations remain the sole responsibility of the authors.

References

[1] José-María Hidalgo Arteaga, Fiodar Hancharou, Florian Thams, and Spyros Chatzivasileiadis. 2019. Deep learning for power system security assessment. In *2019 IEEE Milan PowerTech*. IEEE, 1–6.

[2] Jiyu Huang, Lin Guan, Yinsheng Su, Haicheng Yao, Mengxuan Guo, and Zhi Zhong. 2021. System-scale-free transient contingency screening scheme based on steady-state information: A pooling-ensemble multi-graph learning approach. *IEEE Transactions on Power Systems* 37, 1 (2021), 294–305.

[3] Kip Morison, Lei Wang, and Prabha Kundur. 2004. Power system security assessment. *IEEE power and energy magazine* 2, 5 (2004), 30–39.

[4] Costas Mylonas, Titos Georgoulakis, and Magda Foti. 2024. Facilitating AI and System Operator Synergy: Active Learning-Enhanced Digital Twin Architecture for Day-Ahead Load Forecasting. In *2024 International Conference on Smart Energy Systems and Technologies (SEST)*. IEEE, 1–6.

[5] Zhongtuo Shi, Wei Yao, Yong Tang, Xiaomeng Ai, Jinyu Wen, and Shijie Cheng. 2023. Intelligent power system stability assessment and dominant instability mode identification using integrated active deep learning. *IEEE Transactions on Neural Networks and Learning Systems* 35, 7 (2023), 9970–9984.

A Additional Results

Table 4: Additional KPI summaries and TTT variability across runs.

A.1 Time-to-Target Variability Across Runs

(a) Per-run Time-to-Target (TTT) mean ± std (labels) by strategy. Note: Values here are per-run TTT (mean ± std). The TTT bars in Figures 3–6 are computed on the averaged curve.

Threshold	Strategy	TTT (mean ± std)
ACC ≥ 0.90	entropy	384 ± 207
	margin	384 ± 207
	uncertainty	372 ± 208
	random	440 ± 225
ACC ≥ 0.92	entropy	751 ± 359
	margin	751 ± 359
	uncertainty	751 ± 359
	random	897 ± 318
AUC ≥ 0.97	entropy	432 ± 178
	margin	432 ± 178
	uncertainty	432 ± 178
	random	502 ± 352
AUC ≥ 0.98	entropy	803 ± 304
	margin	803 ± 304
	uncertainty	803 ± 304
	random	1029 ± 386

A.2 Precision/Recall/F1/FNR KPIs and TTT Thresholds

(b) KPIs by strategy for Precision, Recall, F1, and derived FNR. Time-to-Target (TTT) is the number of labels to reach the threshold (e.g., Precision ≥ 0.90, Recall ≥ 0.90; for FNR, TTT corresponds to FNR ≤ 0.10).

Strategy	AULC P	TTT P≥0.90	Final P	AULC R	TTT R≥0.90	Final R	AULC F1	TTT F1≥0.90	Final F1	TTT FNR≤0.10	Final FNR
entropy	0.948	500	0.952	0.916	500	0.922	0.931	500	0.936	500	0.078
margin	0.948	500	0.952	0.916	500	0.922	0.931	500	0.936	500	0.078
random	0.928	960	0.940	0.905	1040	0.906	0.916	1040	0.922	1040	0.094
uncertainty	0.948	500	0.952	0.916	500	0.922	0.931	500	0.936	500	0.078