Christopher Aaron O'Hara

Udacity MSc Capstone – Deep Learning Analysis

February 19, 2026

Overview

This project implements a deep learning intrusion-detection experiment for IIoT/local-cloud telemetry. The task is supervised binary classification (attack vs benign) using a Transformer-based tabular model in PyTorch. The workflow includes a controlled baseline-vs-experiment comparison, additional single-factor ablations, deep ensembles, subgroup risk diagnostics, and reproducibility controls so conclusions are based on measured behavior rather than anecdotal model performance.

Dataset: RT-IoT2022 (UCI Machine Learning Repository, ID 942)

Source link: https://archive.ics.uci.edu/dataset/942/rt-iot2022

Project repository: https://github.com/Ohara124c41/deep-learning-iot-intrusion

Motivation

In local-cloud IIoT environments, telemetry volume and heterogeneity make purely manual triage impractical. A deep model that consistently prioritizes likely attack traffic can reduce analyst latency, but only if the model is trained under explicit controls for imbalance, overfitting risk, and reproducibility (LeCun et al., 2015).

This project matters because operational cybersecurity decisions are sensitive to false negatives (missed attacks) and false positives (alert fatigue). The experiment is therefore designed around multi-metric evaluation, controlled comparisons, and risk-aware interpretation, rather than reporting one headline metric.

Dataset Description

The RT-IoT2022 dataset contains labeled IIoT network-flow records with mixed protocol/service metadata and traffic statistics (UCI Machine Learning Repository, 2024). The raw ingestion in this notebook produced 123,117 rows and 84 columns, and a deterministic runtime-controlled modeling frame of 120,000 rows and 87 columns.

The label column selected from the raw dataset is Attack_type, normalized into a binary target (attack_binary) for triage-oriented detection. The resulting class distribution is imbalanced: 107,829 attack rows and 12,171 benign rows (attack rate 0.898575). The train/validation/test split is 84,000 / 18,000 / 18,000 with 81 numeric features and 2 categorical features, and no split-level label drift was observed in attack-rate checks.

Model Architecture and Design Decisions

Baseline architecture

The baseline model is a Transformer encoder adapted for tabular telemetry: numeric inputs are projected into a token embedding, categorical features are embedded, a learned [CLS] token aggregates representation, and a final linear head outputs attack probability logits. Transformer self-attention was selected because it can model cross-feature interactions without manually specifying pairwise terms (Vaswani et al., 2017).

Training setup

Training uses BCEWithLogitsLoss for binary classification with class weighting (pos_weight) to address label imbalance (He & Garcia, 2009). Optimization uses Adam, mini-batch training, and early stopping to reduce overfitting risk and stabilize convergence (Kingma & Ba, 2015; Prechelt, 1998). Dropout is used for regularization (Srivastava et al., 2014). Reproducibility controls include fixed seeds, fixed split strategy, and deterministic configuration cells.

Experimental Comparison

Controlled comparison design

The required controlled experiment changes exactly one major factor: dropout rate.

- Baseline: dropout 0.10

- Experimental model: dropout 0.30

All other major factors were held constant: data split, feature preprocessing, architecture depth/width, optimizer family, learning rate, epochs, early stopping policy, and metric computation logic. This isolates regularization sensitivity as the primary causal factor.

Additional comparison layer

After the controlled experiment, I ran single-factor ablations (d_model, layer depth, learning rate, optimizer switch to AdamW, class-weight toggle) and two deep-ensemble variants (simple mean and weighted top-3). The AdamW comparison follows decoupled weight-decay practice for modern deep optimization (Loshchilov & Hutter, 2019). A score verifier then selected the final model using a composite validation criterion with operational guardrails on recall and false-positive rate. This extends binary significance-style comparison into a decision-oriented model selection process.

Evaluation Metrics Justification

Because this is imbalanced binary detection, I report multiple complementary metrics rather than accuracy alone.

- F1, precision, recall: summarize attack-detection balance and miss/false-alert tradeoff.

- PR-AUC: emphasized because it is more informative than ROC-AUC under class imbalance (Saito & Rehmsmeier, 2015).

- ROC-AUC: threshold-independent ranking quality (Fawcett, 2006).

- Brier score and RMSE on predicted probabilities: probability quality and calibration-sensitive error (Brier, 1950).

- MCC and balanced accuracy: robust single-number summaries under skewed class distribution (Chicco & Jurman, 2020).

Results

Baseline vs controlled experiment (test set, threshold 0.5)

At threshold 0.5, the baseline model (dropout 0.10) produced accuracy = 0.992056, precision = 0.998818, recall = 0.992333, F1 = 0.995565, ROC-AUC = 0.999424, PR-AUC = 0.999934, Brier = 0.007706, and RMSE(prob) = 0.087782. The controlled experimental model (dropout 0.30) produced accuracy = 0.993833, precision = 0.998263, recall = 0.994868, F1 = 0.996563, ROC-AUC = 0.998971, PR-AUC = 0.999857, Brier = 0.005526, and RMSE(prob) = 0.074334.

Interpretation: the higher-dropout model improved recall, F1, and probability-error metrics (Brier/RMSE), while the baseline retained slightly higher ROC-AUC and precision. For operations where missed attacks are costly, the experiment is preferable at this stage.

Threshold-transfer analysis

Validation-selected thresholds were 0.05 (baseline) and 0.10 (experiment). At these thresholds, baseline reached test F1 = 0.997590 and experiment reached test F1 = 0.997249. This indicates thresholding can materially change deployment behavior and should be treated as part of model policy, not a post-hoc cosmetic change.

Concrete behavior example

Baseline error review showed 124 false negatives and 19 false positives. Most false negatives were ARP_poisioning (120/124), while most false positives were Thing_Speak (17/19). This indicates attack-family-specific blind spots and benign-service confusion that are operationally important even when aggregate metrics are high.

Extended model comparison and final selection

Ablations and ensembles produced several high-performing alternatives. The score verifier selected ensemble_weighted_top3_valf1 as final model with:

- Validation: F1 = 0.997309, PR-AUC = 0.999972, Recall = 0.996661, RMSE(prob) = 0.059295

- Test: F1 = 0.996812, PR-AUC = 0.999961, RMSE(prob) = 0.062383

This final choice reflects a tradeoff-aware objective: near-maximal discrimination with improved probability quality and guardrail compliance.

Subgroup fairness-risk and governance checks

The AIF360 audit selected subgroup is_proto_tcp and returned:

- SPD = -0.305412

- Disparate Impact = 0.671021

- Equal Opportunity Difference = -0.043315

- Average Odds Difference = -0.012872

These values indicate non-trivial subgroup disparity in model behavior and justify ongoing subgroup monitoring rather than assuming uniform performance (Bellamy et al., 2019; Hardt et al., 2016).

NIST-aligned readiness checks passed 6/6, including recall/FPR thresholds, probability quality threshold, reproducibility controls, leakage token scan, and fairness-audit execution.

Interpretation for a Non-Technical Audience

The model is very strong at detecting suspicious traffic in this dataset, but no model is perfect. Some attack families are still missed more often than others, and a few benign traffic patterns can be mistakenly flagged.

The final selected model is better balanced than a single baseline run, and the evaluation process was designed to reduce overconfidence by checking multiple quality measures, subgroup behavior, and operational guardrails. In practice, this model should support analysts, not replace them, and should be monitored continuously as traffic patterns evolve.

Limitations and Risks

Key limitations include dataset shift risk, binary label simplification, and class imbalance. RT-IoT2022 may not fully represent every production IIoT environment, so real deployment performance can differ.

The binary mapping (attack vs benign) hides attack-family granularity, which can mask category-specific failure modes. Metrics are strong in-distribution, but sensitivity to threshold policy and subgroup composition means performance can degrade under traffic drift.

Ethical and Responsible Use

The primary ethical risk is operational misuse through over-automation. If model scores are used for automatic blocking without context, false positives can disrupt legitimate operations and false negatives can allow harmful activity to persist in critical infrastructure contexts.

A second risk is uneven burden across protocol/service groups. The subgroup disparity signals show that not all traffic groups are treated equally. To reduce risk, this workflow keeps human-in-the-loop review for high-impact decisions, requires subgroup-level monitoring, and documents explicit guardrails before deployment.

Reproducibility

The notebook is designed to run top-to-bottom from a clean kernel with deterministic seeds, fixed splits, and explicit configuration. Dependencies are listed in requirements.txt, and dataset fetch/caching is handled programmatically through ucimlrepo plus local cache fallback.

To strengthen rerun consistency, the workflow now exports processed split artifacts to data/processed/ (rtiot2022_binary_rs42_n120000_*): compressed train/validation/test arrays, feature-name files, and a metadata manifest. This allows future experiments to reuse identical prepared inputs without re-deriving preprocessing from raw files.

Future Improvements

Future Work: Deep Learning Extension

The next deep-learning iteration should add true temporal sequence construction per host/service and compare tabular-token Transformer performance against sequence-native architectures under time-aware validation. This would test whether sequential context improves early-stage attack detection beyond the current single-record framing

Future Work: Extension to Generative AI

The most practical generative extension is a retrieval-augmented explanation layer that grounds incident summaries in retrieved evidence (for example, top contributing features, nearest historical incidents, and policy artifacts) instead of free-form generation alone (Lewis et al., 2020). This would convert model outputs into analyst-readable incident narratives while preserving source traceability.

Future Work: Extension to Agentic AI

An agentic layer can orchestrate triage workflows by combining reasoning traces with explicit tool actions (querying telemetry stores, validating policy conditions, proposing containment actions) under human approval gates (Yao et al., 2023). This enables faster, repeatable incident handling while keeping risky actions auditable and reversible.

Future Work: Integrated Industrial Application

For integrated industrial deployment, the architecture should map model and agent controls to OT-specific safeguards, reliability constraints, and safety boundaries defined in current NIST OT security guidance (Stouffer et al., 2023). Practically, that means staged deployment (monitor -> assist -> controlled action), strict separation of recommendation vs

enforcement, and continuous drift/fairness monitoring across plant-specific traffic domains.

Technology and Method Citation Coverage

Technologies used and cited

- Python, NumPy, Pandas, Matplotlib, Seaborn, scikit-learn, PyTorch, Jupyter, UCI dataset tooling (ucimlrepo), AIF360, and NIST CSF guidance (Van Rossum & Drake, 2009; Harris et al., 2020; McKinney, 2010; Hunter, 2007; Waskom, 2021; Pedregosa et al., 2011; Paszke et al., 2019; Project Jupyter, n.d.; Truong, n.d.; Bellamy et al., 2019; National Institute of Standards and Technology, 2024).

Algorithms and methods used and cited

- Transformer self-attention classifier, dropout, BCEWithLogits training, Adam/AdamW optimization, early stopping, deep ensembles, and threshold policy transfer.

- Metrics: precision, recall, F1, ROC-AUC, PR-AUC, MCC, balanced accuracy, Brier score, RMSE(prob).

- Fairness metrics: statistical parity difference, disparate impact, equal opportunity difference, average odds difference.

References

1. Bellamy, R. K. E., Dey, K., Hind, M., Hoffman, S. C., Houde, S., Kannan, K., Lohia, P., Martino, J., Mehta, S., Mojsilovic, A., Nagar, S., Ramamurthy, K. N., Richards, J., Saha, D., Sattigeri, P., Singh, M., Varshney, K. R., & Zhang, Y. (2019). AI Fairness 360: An extensible toolkit for detecting and mitigating algorithmic bias. IBM Journal of Research and Development, 63(4/5), 4:1-4:15. https://doi.org/10.1147/JRD.2019.2942287
2. Brier, G. W. (1950). Verification of forecasts expressed in terms of probability. Monthly Weather Review, 78(1), 1-3. https://doi.org/10.1175/1520-0493(1950)078<0001:VOFEIT>2.0.CO;2
3. Chicco, D., & Jurman, G. (2020). The advantages of the Matthews correlation coefficient (MCC) over F1 score and accuracy in binary classification evaluation. BMC Genomics, 21, 6. https://doi.org/10.1186/s12864-019-6413-7

4. Fawcett, T. (2006). An introduction to ROC analysis. Pattern Recognition Letters, 27(8), 861-874. https://doi.org/10.1016/j.patrec.2005.10.010
5. Hardt, M., Price, E., & Srebro, N. (2016). Equality of opportunity in supervised learning. In Advances in Neural Information Processing Systems 29 (NeurIPS 2016). https://proceedings.neurips.cc/paper/2016/hash/9d2682367c3935defcb1f9e247a97c0d-Abstract.html
6. He, H., & Garcia, E. A. (2009). Learning from imbalanced data. IEEE Transactions on Knowledge and Data Engineering, 21(9), 1263-1284. https://doi.org/10.1109/TKDE.2008.239
7. Hunter, J. D. (2007). Matplotlib: A 2D graphics environment. Computing in Science & Engineering, 9(3), 90-95. https://doi.org/10.1109/MCSE.2007.55
8. Kingma, D. P., & Ba, J. (2015). Adam: A method for stochastic optimization. In International Conference on Learning Representations (ICLR). https://arxiv.org/abs/1412.6980
9. LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. Nature, 521, 436-444. https://doi.org/10.1038/nature14539
10. Loshchilov, I., & Hutter, F. (2019). Decoupled weight decay regularization. In International Conference on Learning Representations (ICLR). https://openreview.net/forum?id=Bkg6RiCqY7
11. Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Kuttler, H., Lewis, M., Yih, W.-t., Rocktaschel, T., Riedel, S., & Kiela, D. (2020). Retrieval-augmented generation for knowledge-intensive NLP tasks. In Advances in Neural Information Processing Systems 33 (NeurIPS 2020). https://arxiv.org/abs/2005.11401
12. McKinney, W. (2010). Data structures for statistical computing in Python. In S. van der Walt & J. Millman (Eds.), Proceedings of the 9th Python in Science Conference (pp. 56-61). https://doi.org/10.25080/Majora-92bf1922-00a
13. National Institute of Standards and Technology. (2024). The NIST Cybersecurity Framework (CSF) 2.0. https://www.nist.gov/cyberframework
14. Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., ... Chintala, S. (2019). PyTorch: An imperative style, high-performance deep learning library. In Advances in Neural Information Processing Systems 32 (NeurIPS 2019). https://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library
15. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. Journal of Machine Learning Research, 12, 2825-2830. https://jmlr.org/papers/v12/pedregosa11a.html

16. Prechelt, L. (1998). Early stopping - but when? In G. B. Orr & K.-R. Muller (Eds.), Neural Networks: Tricks of the Trade (pp. 55-69). Springer. https://doi.org/10.1007/3-540-49430-8_3
17. Saito, T., & Rehmsmeier, M. (2015). The precision-recall plot is more informative than the ROC plot when evaluating binary classifiers on imbalanced datasets. PLOS ONE, 10(3), e0118432. https://doi.org/10.1371/journal.pone.0118432
18. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: A simple way to prevent neural networks from overfitting. Journal of Machine Learning Research, 15, 1929-1958. https://jmlr.org/papers/v15/srivastava14a.html
19. UCI Machine Learning Repository. (2024). RT-IoT2022. University of California, Irvine, School of Information and Computer Sciences. https://archive.ics.uci.edu/dataset/942/rt-iot2022
20. Harris, C. R., Millman, K. J., van der Walt, S. J., Gommers, R., Virtanen, P., Cournapeau, D., Wieser, E., Taylor, J., Berg, S., Smith, N. J., Kern, R., Picus, M., Hoyer, S., van Kerkwijk, M. H., Brett, M., Haldane, A., Fernandez del Rio, J., Wiebe, M., Peterson, P., ... Oliphant, T. E. (2020). Array programming with NumPy. Nature, 585, 357-362. https://doi.org/10.1038/s41586-020-2649-2
21. Project Jupyter. (n.d.). Project Jupyter documentation. https://docs.jupyter.org/en/latest/
22. Truong, P. (n.d.). ucimlrepo: Easily fetch datasets from the UC Irvine Machine Learning Repository. GitHub. https://github.com/uci-ml-repo/ucimlrepo
23. Van Rossum, G., & Drake, F. L. (2009). Python 3 reference manual. CreateSpace. https://docs.python.org/3/
24. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., & Polosukhin, I. (2017). Attention is all you need. In Advances in Neural Information Processing Systems 30 (NeurIPS 2017). https://papers.neurips.cc/paper/7181-attention-is-all-you-need
25. Waskom, M. L. (2021). seaborn: Statistical data visualization. Journal of Open Source Software, 6(60), 3021. https://doi.org/10.21105/joss.03021
26. Stouffer, K. A., Pease, M., Tang, C., Zimmerman, T., Pillitteri, V. Y., Lightman, S., Hahn, A., Saravia, S., Sherule, A., & Thompson, M. (2023). Guide to Operational Technology (OT) Security (NIST SP 800-82r3). National Institute of Standards and Technology. https://doi.org/10.6028/NIST.SP.800-82r3
27. Yao, S., Zhao, J., Yu, D., Du, N., Shafran, I., Narasimhan, K., & Cao, Y. (2023). ReAct: Synergizing reasoning and acting in language models. In International Conference on Learning Representations (ICLR 2023). https://arxiv.org/abs/2210.03629