

Nitin Rathour

5_Thesis-1-83



Research Paper

Document Details

Submission ID

trn:oid:::27535:145311254

Submission Date

Jul 10, 2026, 2:18 PM GMT+5:30

Download Date

Jul 11, 2026, 12:05 AM GMT+5:30

File Name

5_Thesis-1-83.pdf

File Size

10.4 MB

83 Pages

21,686 Words

127,533 Characters

7% Overall Similarity

The combined total of all matches, including overlapping sources, for each database.

Filtered from the Report

- Bibliography
- Small Matches (less than 10 words)

Exclusions

- 1 Excluded Source

Match Groups

-  **66 Not Cited or Quoted 6%**
Matches with neither in-text citation nor quotation marks
-  **16 Missing Quotations 1%**
Matches that are still very similar to source material
-  **0 Missing Citation 0%**
Matches that have quotation marks, but no in-text citation
-  **0 Cited and Quoted 0%**
Matches with in-text citation present, but no quotation marks

Top Sources

- 5%  Internet sources
- 2%  Publications
- 6%  Submitted works (Student Papers)

Integrity Flags

0 Integrity Flags for Review

Our system's algorithms look deeply at a document for any inconsistencies that would set it apart from a normal submission. If we notice something strange, we flag it for you to review.

A Flag is not necessarily an indicator of a problem. However, we'd recommend you focus your attention there for further review.

Match Groups

- 66 Not Cited or Quoted 6%**
Matches with neither in-text citation nor quotation marks
- 16 Missing Quotations 1%**
Matches that are still very similar to source material
- 0 Missing Citation 0%**
Matches that have quotation marks, but no in-text citation
- 0 Cited and Quoted 0%**
Matches with in-text citation present, but no quotation marks

Top Sources

- 5% Internet sources
- 2% Publications
- 6% Submitted works (Student Papers)

Top Sources

The sources with the highest number of matches within the submission. Overlapping sources will not be displayed.

- 1 **Student papers**
Delhi Technological University on 2026-05-23 <1%
- 2 **Student papers**
University of Malakand, Chakdara on 2026-04-09 <1%
- 3 **Student papers**
Delhi Technological University on 2026-05-29 <1%
- 4 **Student papers**
October University for Modern Sciences and Arts (MSA) on 2021-06-15 <1%
- 5 **Student papers**
Delhi Technological University on 2026-06-12 <1%
- 6 **Internet**
arxiv.org <1%
- 7 **Student papers**
Delhi Technological University on 2026-05-07 <1%
- 8 **Student papers**
Higher Education Commission Pakistan on 2026-05-11 <1%
- 9 **Student papers**
University of Monastir on 2026-06-13 <1%
- 10 **Student papers**
Özyegin Üniversitesi on 2026-07-09 <1%

11	Student papers	University of KwaZulu-Natal on 2026-02-02	<1%
12	Student papers	Vellore Institute of Technology on 2026-04-19	<1%
13	Student papers	Delhi Technological University on 2026-05-28	<1%
14	Internet	d-nb.info	<1%
15	Student papers	University of KwaZulu-Natal on 2026-06-25	<1%
16	Internet	f1000research-files.f1000.com	<1%
17	Student papers	University of Wollongong on 2026-05-30	<1%
18	Student papers	University of Malaya on 2026-01-10	<1%
19	Student papers	Delhi Technological University on 2026-05-07	<1%
20	Student papers	The British University in Dubai on 2025-11-20	<1%
21	Internet	edoc.ub.uni-muenchen.de	<1%
22	Internet	123dok.net	<1%
23	Publication	Boram Kim, Guebin Choi. "Non-Euclidean Models for Fraud Detection in Irregular ...	<1%
24	Student papers	University of College Cork on 2016-10-06	<1%

25	Internet	dspace.lib.cranfield.ac.uk	<1%
26	Internet	researchspace.ukzn.ac.za	<1%
27	Internet	trepo.tuni.fi	<1%
28	Internet	www.medrxiv.org	<1%
29	Student papers	Addis Ababa University on 2026-06-09	<1%
30	Publication	Hassiba, Khaled Jamal Eldin. "Ann Framework for Bearing Fault Classification.", H...	<1%
31	Internet	iutoic-dhaka.edu	<1%
32	Internet	uwspace.uwaterloo.ca	<1%
33	Student papers	Islamia College, Peshawar on 2026-06-15	<1%
34	Publication	Qidong Yang, Jonathan Giezendanner, Daniel Salles Civitarese, Johannes Jakubik ...	<1%
35	Student papers	United International College on 2025-12-27	<1%
36	Student papers	Australian National University on 2024-10-24	<1%
37	Student papers	Institute of International Studies on 2026-01-03	<1%
38	Publication	Lafrance, Samantha. "A Multi-Scale Fusion CNN-ViT Hybrid Architecture for Semic...	<1%

39	Publication	Mahmoud A. Ghalwash, Samir Mohamed Abdelrazek, Nabila Hamid Eladawi, Hait...	<1%
40	Student papers	Strathmore University (Main Account) on 2026-07-09	<1%
41	Internet	americaspg.com	<1%
42	Internet	kudos.dfo.no	<1%
43	Internet	www.mdpi.com	<1%
44	Publication	Kamruzzaman, Md.. "An AI-Driven Loan Brokerage Platform: Integrating Socio-Ec...	<1%
45	Student papers	Universiti Tunku Abdul Rahman on 2015-08-28	<1%
46	Student papers	University of Sydney on 2025-05-30	<1%
47	Internet	www.coursehero.com	<1%
48	Internet	www.historytools.org	<1%
49	Student papers	CITY College, University of York Europe Campus on 2025-06-09	<1%
50	Student papers	CSU Northridge on 2025-03-27	<1%
51	Student papers	Delhi Technological University on 2026-05-20	<1%
52	Student papers	Delhi Technological University on 2026-05-21	<1%

53	Publication	Greta Heiser, Aditi Jha, Robert Kelvey, Qimin Huang. "Analysis of the Effect of Sha...	<1%
54	Student papers	Macao Polytechnic Institute on 2025-11-16	<1%
55	Publication	Shirkhani, Ravis. "Automating Test and Analysis of Open and Programmable Radi...	<1%
56	Publication	Vaskar Chakma, Ju Xiaolin, Cao Heling, Xue Feng, Ji Xiaodong, Pan Haiyan, Zhan G...	<1%
57	Publication	Zhu, Hongbo. "Explainable AI in Computer Vision: From CNNs to Transformers.", ...	<1%
58	Internet	api-depositonce.tu-berlin.de	<1%
59	Internet	cfo.economictimes.indiatimes.com	<1%
60	Internet	datafield.dev	<1%
61	Internet	hindutone.com	<1%
62	Internet	khazna.ku.ac.ae	<1%
63	Internet	pure.itu.dk	<1%
64	Internet	thesis.cust.edu.pk	<1%
65	Publication	Md. Sobuj Mia, Sujit Roy, Md Amimul Ihsan, Sadek Hossain, Md. Khabir Uddin Aha...	<1%
66	Publication	Pawar, Ajay. "Testing Scalability of Quantum Transformers Using Quixer.", Steven...	<1%

67	Student papers	
Thesis 2026 on 2026-05-15		<1%
68	Internet	
stax.strath.ac.uk		<1%
69	Internet	
www.preprints.org		<1%

EXPLAINABLE HYBRID FRAMEWORK FOR CREDIT CARD FRAUD DETECTION USING TRANSFORMERS

Thesis Submitted
in Partial Fulfillment of the Requirements for the
Degree of

MASTER OF TECHNOLOGY

in

Data Science

by

Nitin Rathour

(24/DSC/14)

Under the Supervision of

Dr. Sanjay Patidar

Assistant Professor, Department of Software Engineering
Delhi Technological University, Delhi, India



To the
Department of Software Engineering
DELHI TECHNOLOGICAL UNIVERSITY
(Formerly Delhi College of Engineering)
Shahbad Daultpur, Main Bawana Road, Delhi-110042, India

May, 2026

ACKNOWLEDGEMENTS

**Om Krishnaya Vasudevaya Haraye Paramatmane,
Pranata Klesha Nashaya Govindaya Namoh Namah**

My deepest gratitude goes to the divine presence, which has been my constant companion, source of strength, resilience, and guidance throughout my academic journey. This guiding force has given me the strength to face the complexities of this academic journey, both in times of difficulty and success, keeping me grounded and patient in moments of uncertainty.

I would like to thank my supervisor **Dr. Sanjay Patidar** for his guidance, support and suggestions throughout this research. The quality and direction of this work has been greatly influenced by his mentorship and encouragement, and his confidence in my abilities pushed me to hold this work to a higher standard.

I would like to thank **Prof. Ruchika Malhotra**, Head of the Department, for her encouragement and providing an environment for innovation in the field of academics. I would also like to express my sincere gratitude to **Dr. Abhilasha Sharma** for her constant support, guidance and encouragement during this work, which helped me navigate several difficult phases of this research. I would like to express my gratitude to **Delhi Technological University** for providing me the resources and an inspiring academic atmosphere to complete this research successfully. I also thank the faculty members, colleagues and peers for their cooperation and support.

I would like to thank my parents for their unconditional love, support and encouragement, which has always been my greatest strength. I am also deeply grateful to my **grandparents** for their blessings, wisdom and unwavering belief in me. I also thank my family and friends for their constant motivation during challenging times and for reminding me to maintain balance and perspective. This thesis would not have been possible without the support and encouragement of everyone who has been involved in this journey. I am so thankful to all of them.

Nitin Rathour
24/DSC/14



DELHI TECHNOLOGICAL UNIVERSITY
(Formerly Delhi College of Engineering)
Shahbad Daulatpur, Main Bawana Road, Delhi-42

CANDIDATE’S DECLARATION

I Nitin Rathour hereby certify that the work which is being presented in the thesis entitled Explainable Hybrid Framework for Credit Card Fraud Detection Using Transformers in partial fulfillment of the requirements for the award of the Degree of Master of Technology in Data Science, submitted in the Department of Software Engineering, Delhi Technological University is an authentic record of my own work carried out during the period from August 2024 to May 2026 under the supervision of Dr. Sanjay Patidar.

The matter presented in the thesis has not been submitted by me for the award of any other degree of this or any other Institute.

Candidate’s Signature

This is to certify that the student has incorporated all the corrections suggested by the examiners in the thesis and the statement made by the candidate is correct to the best of our knowledge.

Signature of Supervisor(s)

Signature of External Examiner



DELHI TECHNOLOGICAL UNIVERSITY

(Formerly Delhi College of Engineering)

Shahbad Daultpur, Main Bawana Road, Delhi-42

CERTIFICATE BY THE SUPERVISOR

3 Certified that **Nitin Rathor** (24/DSC/14) has carried out their search work presented in this thesis entitled “**Explainable Hybrid Framework for Credit Card Fraud Detection Using Transformers**” for the award of **Master of Technology** from Department of Software Engineering, Delhi Technological University, Delhi, under my supervision. The thesis embodies results of original work, and studies are carried out by the student herself and the contents of the thesis do not form the basis for the award of any other degree to the candidate or to anybody else from this or any other University/Institution.

Place: Delhi

Date: May 21, 2026

Dr. Sanjay Patidar
Assistant Professor
Department of Software Engineering
Delhi Technological University
Delhi, India

Explainable Hybrid Framework for Credit Card Fraud Detection Using Transformers

Nitin Rathour

ABSTRACT

Credit card fraud causes huge losses for banks and customers every year. The problem is hard for three reasons: fraud is extremely rare compared to normal transactions, fraud patterns change over time, and regulators now require automated decisions to be explainable. Global card fraud reached \$33.41 billion in 2024 and is projected to cross \$403.88 billion over the next decade. This work proposes a single hybrid pipeline to address all three problems of the ULB Credit Card Fraud benchmark containing 284 807 transactions and just 492 fraud cases (a 1:578 class ratio). The pipeline has six connected stages. A Taguchi L_{25} search picks a Quantile-then-Robust scaling pair, giving a five-fold cross-validated AUC of 0.97940 ± 0.00564 . A Conditional Tabular GAN trained only on the 344 real fraud rows generates 132 336 synthetic fraud samples, lifting the training set to a 40% fraud ratio. An autoencoder trained on legitimate transactions alone produces a reconstruction-error signal that separates fraud from non-fraud by a factor of 64 on the test set. A two-track design feeds augmented data to the tabular models and real-only chronological data to the sequence model. Three structurally different base learners, a Time-Aware Transformer (282,881 parameters), XGBoost on 64 leakage-free features, and an Explainable Boosting Machine, are combined through rank averaging. Platt scaling cuts test Expected Calibration Error from 0.498 to 0.0018, a 281-fold improvement, and three independent explainability methods (EBM shape functions, TreeSHAP, and LIME) unanimously rank autoencoder reconstruction error as the top predictor. On the held-out test set the system reaches accuracy of 0.9994, AUC-ROC of 0.9764, AUC-PR of 0.8244 (about 476 times the random baseline), F_1 of 0.8088, precision of 0.8871, recall of 0.7432, and Matthews correlation coefficient of 0.8117 at the F_1 -maximising threshold; a high-recall operating point catches 86.5% of fraud while flagging only 1.6% of legitimate transactions. The work extends the author's earlier ICCCNT 2025 conference work along five dimensions: generative oversampling, sequential modelling, glass-box interpretability, formal probability calibration, and three-method explainability cross-validation.

Keywords: Credit Card Fraud Detection, Deep Learning, Explainable AI, Sequential Modeling, Generative Oversampling, CTGAN, Transformer, Autoencoder, Ensemble Learning, Anomaly Detection.

LIST OF PUBLICATIONS

International Conference

1. Rathour, N. and Patidar, S., 2025. A threshold-tuned hybrid ensemble for credit card fraud detection. In *Proceedings of the 16th International Conference on Computing, Communication and Networking Technologies (ICCCNT 2025)* (IIT Indore, Madhya Pradesh, India, 6–11 July 2025). (IEEE, Scopus-Indexed).

TABLE OF CONTENTS

Acknowledgement	i
Candidate's Declaration	ii
Certificate	iii
Abstract	iv
List of Publications	v
Table of Contents	vi
List of Table(s)	x
List of Figure(s)	xii
List of Symbols, Abbreviations and Nomenclature	xiii
1 Introduction	1
1.1 Background and Motivation	1
1.2 Three Interlocking Technical Difficulties	1
1.3 Problem Statement, Objectives, and Research Questions	3
1.4 Contributions of the Thesis	4
1.5 Pipeline Overview	6
1.6 Organisation of the Thesis	7
1.7 Summary	7
2 Literature Review	9
2.1 Introduction	9
2.2 Class-Imbalance Handling Techniques	9
2.3 Deep Learning for Fraud Detection	11
2.4 Explainable Artificial Intelligence in Finance	12
2.5 Critical Analysis of Foundational Studies	13
2.6 Research Gaps and Mapping to work Contributions	15
2.7 Summary	16
3 Theoretical Background	18
3.1 Introduction	18
3.2 Conditional Tabular GAN	18

3.3	Autoencoder and Reconstruction-Error Anomaly Scoring	20
3.4	Self-Attention and the Transformer Encoder	20
3.5	Gradient Boosting and the Explainable Boosting Machine	21
3.6	Probability Calibration	22
3.7	Shapley Values and Local-Linear Approximation	23
3.8	Summary	24
4	Proposed Methodology	25
4.1	Overview of the Proposed Framework	25
4.2	Data Pre-processing	27
4.2.1	ULB Dataset	27
4.2.2	Temporal Feature Engineering	27
4.2.3	Taguchi L_{25} Scaler Search	27
4.2.4	Train-Validation-Test Split	28
4.3	Class-Imbalance Handling	28
4.3.1	CTGAN Configuration	28
4.3.2	Synthetic Sample Size	29
4.3.3	Class-Weighted Loss	29
4.4	Two-Track Architecture and Feature Engineering	30
4.4.1	The Two-Track Construction	30
4.4.2	Engineered Temporal Features	30
4.4.3	Cyclical Hour Encoding	30
4.4.4	Autoencoder Pre-training	30
4.4.5	Sequence Construction	32
4.5	Hybrid Base Learners	32
4.5.1	Time-Aware Transformer	33
4.5.2	XGBoost with Leakage Mitigation	33
4.5.3	Explainable Boosting Machine	35
4.6	Rank-Averaged Ensemble and Calibration	36
4.6.1	Rank Averaging	36
4.6.2	Platt Calibration	36
4.6.3	Threshold Tuning	37
4.7	Explainability Framework	37
4.8	Reproducibility	37
4.9	Summary	38
5	Implementation	39
5.1	Introduction	39

5.2	Hardware and Software Stack	39
5.3	Reproducibility Discipline	40
5.4	Three Critical Engineering Decisions	41
5.4.1	Mixed-precision training for the sequence learner	41
5.4.2	Lazy sequence loading: saving 2.9 GB	42
5.4.3	The temporal-leakage discovery	42
5.5	Pipeline Orchestration and Component Persistence	43
5.6	The Verification Dossier	44
5.7	Engineering Trade-offs	44
5.8	Summary	45
6	Results and Discussion	46
6.1	Headline Performance	46
6.2	Pre-processing Results: Taguchi L_{25} Search	47
6.3	CTGAN Augmentation Results	49
6.4	Autoencoder Reconstruction-Error Analysis	49
6.5	Per-Model and Ensemble Performance	51
6.6	Calibration, Thresholds, and Explainability	53
6.6.1	Calibration Quality	53
6.6.2	Threshold-Strategy Outcomes	54
6.6.3	Three-Method Explainability Cross-Validation	55
6.7	Answers to Research Questions	55
6.8	Comparison with Foundational Studies	56
6.9	Summary	57
7	Conclusions and Future Work	59
7.1	Synwork of Findings	59
7.2	Principal Contributions	60
7.3	Limitations	60
7.4	Future Work	61
7.5	Summary	62
	References	63
	List of Publications and their Proofs	67
	Plagiarism Report	71
	Curriculum Vitae	72

LIST OF TABLE[s]

1.1	Aspect-wise comparison of the author’s earlier conference work [1] and the present work.	6
2.1	Comparative analysis of class-imbalance handling techniques in the fraud-detection literature.	10
2.2	Comparative summary of seven foundational fraud-detection studies and the present work. AUC-ROC values are reported on benchmark splits; “Calib?” indicates whether formal probability calibration is reported.	14
4.1	Stratified 70/15/15 split of the ULB dataset.	28
4.2	CTGAN training configuration.	29
4.3	Reductions from the original specification.	33
5.1	Principal software dependencies, pinned versions, and roles in the integrated system.	40
5.2	Four engineering trade-offs adopted in the present implementation, with their costs, benefits, and conditional applicability.	45
6.1	Headline test-set performance of the full pipeline. Test partition contains 42 722 transactions of which 74 are fraudulent.	47
6.2	Top five and bottom five scaler configurations from the Taguchi L_{25} search. The 0.013 spread between best and worst configurations exceeds the cross-fold standard deviation by roughly a factor of two.	48
6.3	CTGAN augmentation configuration and post-augmentation class statistics.	49
6.4	Median autoencoder reconstruction error ϵ by partition and class. The $48\text{--}76\times$ class separation across all partitions confirms a reliable unsupervised fraud signal that generalises beyond the training data.	51
6.5	Per-model and ensemble performance on the held-out test partition. EBM achieves the highest single-model AUC-ROC; the rank-averaged ensemble achieves the highest AUC-PR.	51
6.6	Pairwise Spearman rank correlations between base-learner predictions on the test partition. Values substantially below the ~ 0.95 typical of near-collinear models confirm structural diversity.	52
6.7	Calibration quality before and after Platt scaling. The $281\text{--}285\times$ ECE improvement is consistent across validation and test partitions, indicating that the calibration generalises robustly.	53

6.8	Threshold-strategy outcomes on the test partition. The F_1 -maximising and PR G -mean strategies converge to the same threshold (0.1957), reflecting a sharp “knee” in the validation PR curve at this point. . . .	54
6.9	Cross-method feature-attribution comparison on the test partition. The autoencoder reconstruction error is unanimously ranked top by all three methods.	55
6.10	Comparison of present results against foundational studies. The present work achieves competitive AUC-ROC and F_1 alongside two methodological extensions absent from prior studies: explicit AUC-PR reporting and formal probability calibration.	57
7.1	Future-work programme: six directions extending the present work, each addressing a specific limitation identified in Section 7.3.	61

LIST OF FIGURE[s]

1.1	The three interlocking technical difficulties of practical credit-card fraud detection: statistical imbalance, temporal sequence structure, and regulatory explainability.	2
1.2	Mapping of the five identified research challenges (left) to the five contributions of this work (right).	4
1.3	End-to-end pipeline of the proposed hybrid interpretable fraud-detection framework. Raw transactions flow through pre-processing, generative augmentation, two-track feature engineering, hybrid ensemble training, and post-processing for calibration and explainability.	7
2.1	Generational evolution of credit-card fraud-detection methodology. The present work sits within the fourth generation, integrating multi-paradigm modelling with formal calibration and three-method explainability.	10
2.2	Mapping of the five research gaps identified through literature review (left) to the five corresponding solutions introduced in this work (right).	15
3.1	The seven theoretical pillars supporting the proposed methodology, grouped by their functional role in the pipeline	19
4.1	Two-track data architecture. Synthetic samples augment the tabular track only; the sequence track preserves real chronological structure required by the Transformer.	31
4.2	Symmetric autoencoder with 32-dimensional bottleneck. The reconstruction error $\varepsilon = \ x - g_\phi(f_\theta(x))\ _2^2$ serves as an anomaly feature for the gradient-boosted base learners.	32
4.3	Time-Aware Transformer encoder. Inter-arrival times Δt are concatenated as input features rather than encoded as a custom attention bias	34
4.4	Rank-averaged ensemble with Platt calibration. Three structurally diverse base learners produce probabilities that are converted to ranks, equally averaged, and calibrated to true fraud probabilities.	35
5.1	Component dependency graph across the six output directories. Each downstream component can resume from the persisted outputs of its predecessor without rerunning upstream cells, a pattern that compresses iteration cycles during methodology development.	43

6.1	Taguchi L_{25} heatmap of five-fold cross-validated AUC-ROC across all twenty-five scaler-pair combinations. The selected pipeline (Quantile $\rightarrow$ Robust, marked $\star$) achieves AUC = 0.97940. Rows correspond to the first scaler in the pair; columns to the second.	48
6.2	Autoencoder training and validation MSE across thirty epochs. The smooth monotonic decrease and small train-validation gap confirm stable, generalising training. Best validation MSE of 0.003398 is achieved at epoch 30.	50
6.3	Test-partition reconstruction-error distribution by class. The minimal overlap between legitimate (median 0.11) and fraud (median 6.87) distributions confirms that ae_error carries strong discriminative signal independent of any supervised classifier — a $64\times$ median class separation.	50
6.4	Time-Aware Transformer training and validation curves across twenty epochs. The validation AUC-ROC peaks at 0.98219 at epoch 8 (marked); subsequent epochs show overfitting and trigger early stopping.	51
6.5	(a) Test-partition precision-recall curves for the three base learners and the rank-averaged ensemble. The ensemble achieves the highest AUC-PR at 0.82441, exceeding the random baseline by approximately $476\times$, (b) Test-partition precision-recall curves for the three base learners and the rank-averaged ensemble. The ensemble achieves the highest AUC-PR at 0.82441, exceeding the random baseline by approximately $476\times$	52
6.6	Compare test-partition reliability diagram before and after Platt scaling. The calibrated curve is very close to the diagonal of perfect calibration, while the raw scores are very far from the diagonal.	53
6.7	Confusion matrix for test set using three threshold schemes. The F_1 -max and PR- G -mean strategies coincide; the high-recall strategy trades precision for recall.	54

LIST OF SYMBOLS, ABBREVIATIONS AND NOMENCLATURE

Symbols	Description
α	Significance level / weight coefficient
β	Logistic regression coefficient (Platt scaling)
ε	Autoencoder reconstruction error
η	Learning rate
θ, ϕ	Trainable model parameters (encoder, decoder)
λ	Gradient-penalty coefficient (WGAN-GP)
σ	Sigmoid activation function
τ	Decision threshold
$\tau_{F_1}^*$	F_1 -maximising threshold
$\tau_{R \geq 0.9}^*$	High-recall threshold (recall ≥ 0.90)
ϕ_j	Shapley value of feature j
ρ	Class prevalence (fraud rate)
Δt	Inter-arrival time between transactions
$\hat{p}$	Predicted probability of fraud
$\hat{p}_{cal}$	Calibrated probability after Platt scaling
$\hat{y}$	Predicted class label
∇	Gradient operator
$\mathbb{E}$	Expectation operator
$\mathcal{L}$	Loss function
Abbreviations	Full Form
AE	Autoencoder
AdamW	Adam with decoupled weight decay (optimiser)
AUC	Area Under the Curve
AUC-PR	Area Under the Precision-Recall Curve
AUC-ROC	Area Under the Receiver Operating Characteristic Curve
BCE	Binary Cross Entropy
CCNN	Continuous-Coupled Neural Network
CNN	Convolutional Neural Network
CPU	Central Processing Unit
CTGAN	Conditional Tabular Generative Adversarial Network
CV	Cross-Validation
DTU	Delhi Technological University

EBM	Explainable Boosting Machine
ECE	Expected Calibration Error
EDA	Exploratory Data Analysis
FFN	Feed-Forward Network
FN	False Negative
FP	False Positive
FPR	False Positive Rate
GA ² M	Generalised Additive Model with Interactions
GAN	Generative Adversarial Network
GBDT	Gradient Boosted Decision Trees
GDPR	General Data Protection Regulation
GELU	Gaussian Error Linear Unit
GPU	Graphics Processing Unit
IF	Isolation Forest
KNN	K-Nearest Neighbours
LightGBM	Light Gradient Boosting Machine
LIME	Local Interpretable Model-agnostic Explanations
LSTM	Long Short-Term Memory
MCC	Matthews Correlation Coefficient
ML	Machine Learning
MLP	Multi-Layer Perceptron
MSE	Mean Squared Error
NPCI	National Payments Corporation of India
PCA	Principal Component Analysis
PR	Precision-Recall
RBI	Reserve Bank of India
RBM	Restricted Boltzmann Machine
ReLU	Rectified Linear Unit
RF	Random Forest
RNN	Recurrent Neural Network
ROC	Receiver Operating Characteristic
RQ	Research Question
SHAP	SHapley Additive exPlanations
SMOTE	Synthetic Minority Over-sampling Technique
SVM	Support Vector Machine
TN	True Negative

TP	True Positive
TPR	True Positive Rate
TX	Transformer
ULB	Universit'e Libre de Bruxelles (dataset source)
UPI	Unified Payments Interface
WGAN-GP	Wasserstein GAN with Gradient Penalty
XAI	Explainable Artificial Intelligence
XGBoost	Extreme Gradient Boosting
Nomenclature	Meaning
x_i	Feature vector of the i -th transaction
y_i	Binary fraud label of the i -th transaction
z	Latent representation from autoencoder bottleneck
d	Input feature dimension
d_{model}	Transformer model embedding dimension
d_k, d_v	Key and value dimensions in attention
h	Number of attention heads
k	Bottleneck dimension of autoencoder
n	Sequence window length (50 transactions)
N	Total number of samples in a partition
L	Number of legitimate transactions
F	Number of real fraud transactions
S	Number of synthetic samples generated by CTGAN
A, B	Platt scaling parameters (slope, intercept)
G_θ	Generator network of CTGAN
D_ϕ	Discriminator (critic) network of CTGAN
f_θ	Encoder function of autoencoder
g_ϕ	Decoder function of autoencoder
f_j	Per-feature shape function in EBM
$f_{j,k}$	Pairwise interaction shape function in EBM
Q, K, V	Query, Key, Value matrices in attention
W_Q, W_K, W_V	Learned projection matrices for Q, K, V
W_O	Output projection matrix in multi-head attention
$\mathcal{D}_{\text{train}}$	Training partition (199,364 rows)
$\mathcal{D}_{\text{val}}$	Validation partition (42,721 rows)
$\mathcal{D}_{\text{test}}$	Test partition (42,722 rows)
$\mathcal{D}_{\text{tab}}$	Augmented tabular track (331,700 rows)

$\mathcal{D}_{\text{seq}}$	Real-only sequence track (199,364 rows)
$V_1, \dots, V_{28}$	PCA-derived features in ULB dataset
ae_error	Autoencoder reconstruction-error feature
Hour	Hour-of-day feature ($\lfloor \text{Time}/3600 \rfloor \bmod 24$)
Hour _{sin} , Hour _{cos}	Cyclical sine-cosine encoding of Hour

CHAPTER 1

INTRODUCTION

1.1 Background and Motivation

Worldwide card-payment volume totaled \$51.92 trillion in 2024 for all credit, debit and prepaid cards that were issued via global, regional and private-label networks. The same development has increased the "attack surface" for financial criminals. According to the Nilson Report, the total cost of card fraud losses globally is expected to reach nearly \$403.88 billion over the next ten years if technology and regulation stay the same. If technology and regulation remain unchanged, global card fraud losses will be nearly \$403.88 billion over the next decade, according to the Nilson Report. It is worthy of comment that the geographic asymmetry exists. The United States makes up about a quarter of the total global amount of fraud losses, while at the same time accounting for nearly half the total number of card-not-present transactions, reflecting the slow adoption of more robust authentication methods here [2].

This warrants some special consideration in the Indian context. NPCI launched the Unified Payments Interface in 2016. From roughly eighteen million transactions in its first year, UPI has scaled to approximately 186 billion transactions worth 261 lakh crore during the 2024–25 financial year [3]. India now handles nearly half of all real-time digital-payment transactions worldwide, exceeding the combined volume of the next four countries [4]. The Reserve Bank of India's Annual Report for 2023–24 reports a fivefold year-on-year jump in the value of digital-payment fraud, which rose to 14.57 billion in the financial year ending March 2024 [5]. Card and internet-banking fraud cases climbed from 6,699 in 2022–23 to 29,082 in 2023–24, a 334 per cent increase in a single financial year [5].

1.2 Three Interlocking Technical Difficulties

Practical fraud detection sits at the intersection of three difficulties that have proved persistently resistant to single-method solutions. Figure 1.1 maps each difficulty to the methodological demands it places on any working detection system.

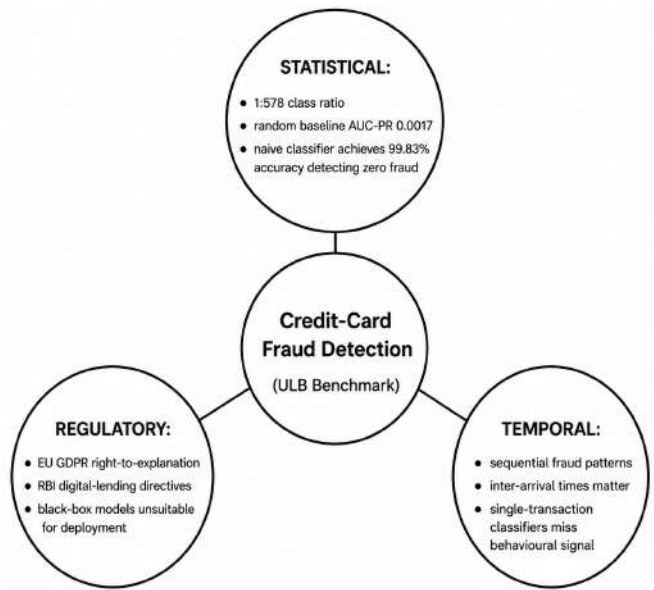


Figure 1.1: The three interlocking technical difficulties of practical credit-card fraud detection: statistical imbalance, temporal sequence structure, and regulatory explainability.

The first difficulty is *statistical*. Fraud is a rare event. In the benchmark European cardholder dataset released by the Machine Learning Group of the Université Libre de Bruxelles, only 492 of 284 807 transactions carry the fraud label, a ratio of roughly 1:578 [6, 7]. A trivial classifier that predicts “legitimate” for every input still achieves 99.83 per cent accuracy on this benchmark, all while detecting zero fraudulent transactions. Useful classifiers therefore have to be evaluated on metrics built for imbalanced settings, including the area under the precision-recall curve [8, 9] and the Matthews correlation coefficient [10, 11].

The second difficulty is *temporal*. Fraud rarely occurs in isolation. Compromised credentials are typically used in sequences of related transactions, often within minutes of one another. Inter-arrival times, deviations from historical norms, and rolling-window statistics all carry behavioural signal that single-transaction classifiers miss. Architectures that directly model sequences are desirable for capturing this signal: attention-based encoders [12] or recurrent networks. Architectures that directly model sequences are desirable for capturing this signal: attention-based encoders [12] or recurrent networks.

The third problem is *regulatory*. Automated decisions with an impact on consumers are increasingly under scrutiny from financial regulators in key jurisdictions such as the European Union, the United States and India, which are requiring that

such decisions be auditable and explainable [?]. The EU's General Data Protection Regulation (GDPR) also provides a right to meaningful information on automated decisions in Article 22, and the Reserve Bank of India's master directions on digital lending also have similar requirements for transparency. When models fail to provide explainable reasons per decision, they are hard to deploy in production, despite being superior at decision making to the explainable models. Much work that is written on explainable AI to date can be traced back to this tension, such as [13–15].

The author's earlier work, presented at the 16th IEEE International Conference on Computing, Communication and Networking Technologies held at IIT Indore in July 2025, addressed a subset of these difficulties. That paper proposed a threshold-tuned hybrid ensemble combining Random Forest, an unsupervised autoencoder, and Isolation Forest [1]. It deliberately omitted synthetic oversampling on grounds that SMOTE-based augmentation might inject artefacts inconsistent with real fraud distributions [16, 17], and recorded 0.96 precision and 0.81 recall on the ULB test set. Three gaps were left unaddressed. The earlier work used neither generative oversampling nor sequential modelling. Its probability scores were uncalibrated. And its feature attributions came from a single post-hoc method.

1.3 Problem Statement, Objectives, and Research Questions

The problem this work addresses is stated formally as follows. Given a stream of card-based transactions $\mathcal{T} = \{x_1, x_2, \dots, x_T\}$, with each $x_i \in \mathbb{R}^d$ accompanied by a binary label $y_i \in \{0, 1\}$ (legitimate or fraudulent), the goal is to learn a function $f : \mathbb{R}^d \rightarrow [0, 1]$ whose output ranks fraudulent transactions ahead of legitimate ones *and* reports a calibrated probability of fraud usable in cost-sensitive operational decisions.

Two operational features set this problem apart from generic binary classification. The positive-class prevalence $\rho = \mathbb{E}[y_i]$ is small, typically below one per cent. The impact of false negative (missed fraud) vs false positive (incorrectly flagged legitimate transactions) is significant, with the loss of financial gain if a fraud goes undetected, and the costs to investigate each one increasing with count of alerts raised.

Five research objectives are pursued, one for each gap left by the prior conference paper. They are: introducing generative oversampling through a Conditional Tabular GAN [18]; adding sequential context through a Time-Aware Transformer [12]; substituting an Explainable Boosting Machine [15] for the Random-Forest component; calibrating ensemble probabilities through Platt scaling [19] with Brier score [20] and

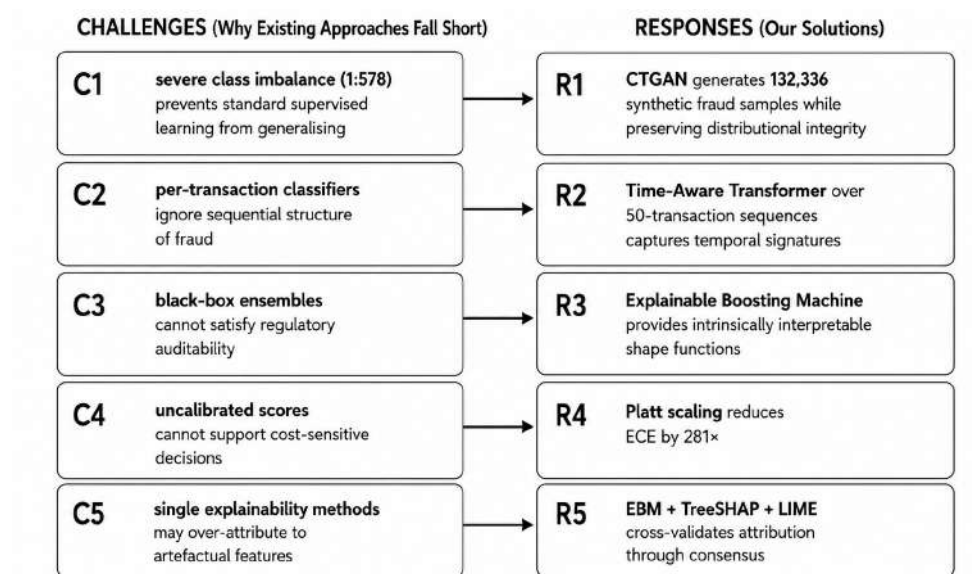


Figure 1.2: Mapping of the five identified research challenges (left) to the five contributions of this work (right).

Expected Calibration Error [21] as diagnostics; and cross-validating feature attributions across three independent explainability methods.

These objectives translate into four research questions:

RQ1 — Can CTGAN-based generative oversampling improve test-set fraud-detection performance compared to a no-oversampling threshold-tuned baseline?

RQ2 — Does adding a Time-Aware Transformer to a tabular classifier ensemble produce diversity-driven gains?

RQ3 — Does post-hoc Platt scaling applied to the rank-averaged ensemble produce probability estimates suitable for cost-sensitive decisions?

RQ4 — Do three independent feature-attribution methods converge on the same dominant predictor?

1.4 Contributions of the Thesis

This work advances the author’s earlier conference work [1] along five concrete dimensions. Figure 1.2 maps the five challenges that limited the prior framework to the five contributions introduced here.

Contribution 1 (R1): Generative oversampling. The conference paper omitted oversampling to preserve transactional integrity. The present work revisits that decision through a Conditional Tabular GAN [18] trained exclusively on the 344 real fraud rows of the training partition. The trained generator yields 132 336 synthetic fraud samples, lifting the training set to a 40 per cent fraud ratio. Synthetic data is restricted to the augmented tabular track and explicitly excluded from the sequential-modelling track, preserving the causal time-semantics required by the Transformer encoder.

Contribution 2 (R2): Time-Aware Transformer. The conference paper relied entirely on per-transaction learners. The present work introduces a Transformer encoder [12] that ingests sequences of fifty consecutive transactions, each concatenated with its inter-arrival time gap. The architecture uses sinusoidal positional encoding, two pre-LayerNorm encoder layers with four attention heads each, GELU activation, and last-token pooling.

Contribution 3 (R3): Explainable Boosting Machine. The conference paper used Random Forest [22] with post-hoc feature-importance scores. The present work substitutes an Explainable Boosting Machine [15] that produces intrinsically interpretable per-feature shape functions, trading a small predictive penalty for direct auditability of the model's decision logic.

Contribution 4 (R4): Probability calibration. The conference paper reported uncalibrated weighted scores. The present work applies Platt scaling [19] to the rank-averaged ensemble, cutting test-set Expected Calibration Error from 0.498 to 0.0018 — a 281-fold improvement that brings the mean predicted probability to within four decimal places of the empirical fraud rate.

Contribution 5 (R5): Three-method explainability. Three structurally distinct attribution methods are applied to the trained ensemble: EBM shape functions [15], TreeSHAP applied to XGBoost [13, 23], and LIME applied to the calibrated full ensemble [14]. All three methods unanimously identify the autoencoder reconstruction error as the top-ranked feature, with substantial overlap among the top-ten features identified by each method.

Table 1.1 sets out the dimensions along which this work extends the conference paper.

Table 1.1: Aspect-wise comparison of the author’s earlier conference work [1] and the present work.

Aspect	ICCCNT 2025 (prior work)	Present work (extension)
Imbalance handling	None (deliberate SMOTE-free design)	CTGAN trained on 344 real fraud rows; 132336 synthetic samples added
Sequential modelling	Per-transaction classifiers only	Time-Aware Transformer over sequences of 50 transactions with Δt gaps
Principal supervised model	Random Forest (post-hoc importance)	Explainable Boosting Machine (intrinsically interpretable)
Probability calibration	None (raw weighted scores)	Platt scaling; ECE reduced $0.498 \rightarrow 0.0018$ ($281\times$)
Explainability analysis	None	Three independent methods (EBM, TreeSHAP, LIME) with cross-method consensus
Headline operating point	$F_1 = 0.88$ at $\tau \approx 0.385$	$F_1 = 0.8088$ at $\tau = 0.1957$ (calibrated)

1.5 Pipeline Overview

The complete system threads five methodological stages into a single end-to-end pipeline. Raw transaction data is first characterised through exploratory analysis. Pre-processing then applies a Taguchi-tuned scaling pipeline [24]. Synthetic fraud samples are generated through CTGAN [18]. Temporal features are engineered, an autoencoder is trained for unsupervised representation learning, and a two-track architecture separates the augmented tabular track from the real-only sequence track. Three base learners Time-Aware Transformer, XGBoost, and EBM are trained and combined through a rank-averaged ensemble [25]. The ensemble is calibrated, thresholds are tuned across three operational regimes, and three independent explainability artefacts are produced. Figure 1.3 illustrates the end-to-end flow.

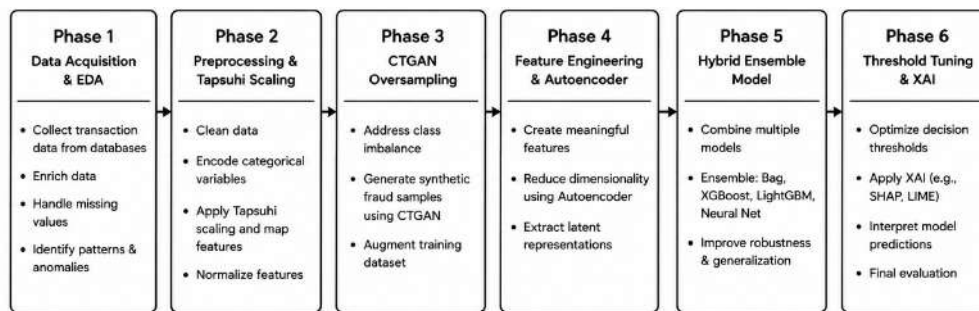


Figure 1.3: End-to-end pipeline of the proposed hybrid interpretable fraud-detection framework. Raw transactions flow through pre-processing, generative augmentation, two-track feature engineering, hybrid ensemble training, and post-processing for calibration and explainability.

1.6 Organisation of the Thesis

The remainder of this work is structured into six chapters. Chapter 2 reviews the literature on class-imbalance handling, deep learning for fraud detection, explainable artificial intelligence, and ensemble methods, analysing seven foundational studies and articulating the gaps this work addresses. Chapter 3 develops the theoretical background, covering the WGAN-GP training objective [26, 27], autoencoder-based anomaly detection [28, 29], self-attention [12], gradient boosting [23, 30], generalised additive models [15, 31], probability calibration [19, 32], and the Shapley-value framework [33]. Chapter 4 sets out the proposed methodology in unified form, including the two-track data architecture motivated by the absence of a card identifier in ULB. Chapter 5 documents the software-engineering aspects: hardware and software environment, reproducibility configuration, and key implementation choices including mixed-precision training [34]. Chapter 6 reports the experimental results, beginning with the Taguchi search and proceeding through CTGAN training behaviour, autoencoder reconstruction-error analysis, per-model and ensemble performance, calibration quality, threshold-strategy outcomes, and the three-method explainability comparison. The contributions are summarized in Chapter 7; it highlights the limitations found, and identifies future directions, such as extending this work to the IEEE-CIS dataset [35].

1.7 Summary

It has focused the problem of credit card fraud detection through the lens of three perennial technical challenges: severe statistical imbalance (a 1:578 ratio of frauds to non-frauds in the ULB benchmark [6, 7]; this imbalance is frequent in many areas of pattern recognition), temporal sequence structure which single-transaction classifiers

cannot leverage, and regulatory calls for auditable and explainable automated decisions [36]. The Indian context sharpens these challenges: the digital-payment fraud values rose fivefold in a single financial year [5], and India now handles nearly half of all real-time digital-payment transactions worldwide [4].

The author's earlier conference work [1] addressed a subset of these difficulties through a threshold-tuned hybrid ensemble of Random Forest, an unsupervised autoencoder, and Isolation Forest. Three gaps were left open: that work used neither generative oversampling nor sequential modelling; its probability scores were uncalibrated; and its feature attributions came from a single post-hoc method.

The present work advances this earlier work along five concrete dimensions, one for each identified gap: generative oversampling through a Conditional Tabular GAN [18]; sequential modelling through a Time-Aware Transformer [12]; intrinsic interpretability through an Explainable Boosting Machine [15] as the principal supervised component; formal probability calibration through Platt scaling [19] with Brier score [20] and Expected Calibration Error [21] diagnostics; and three-method explainability cross-validation combining EBM shape functions, TreeSHAP [13], and LIME [14]. Each contribution has methodological precedent in the literature reviewed in Chapter 2; the integration of all five within a single pipeline is, to the author's knowledge, novel.

Four research questions follow from these contributions: whether CTGAN-based oversampling improves test-set performance over a no-oversampling baseline; whether the Transformer contributes diversity-driven ensemble gains; whether Platt scaling produces calibration quality suitable for cost-sensitive deployment; and whether three independent explainability methods converge on the same dominant predictor. The remaining chapters develop the methodology in detail and answer each question empirically. Chapter 2 reviews the literature on imbalance handling, deep learning for fraud detection, explainable AI, and ensemble methods. Chapter 3 develops the theoretical background. Chapter 4 presents the proposed methodology in unified form. Chapter 5 documents the software-engineering aspects. Chapter 6 reports the experimental results. The contributions are summarized in Chapter 7 and future trends are outlined such as the extension to the IEEE-CIS dataset [35].

CHAPTER 2

LITERATURE REVIEW

There are four broad generations of credit-card fraud detection research in the last 20 years. Every generation had to deal with some aspects of their forebears' weaknesses and innovated with new tensions of their own. This development from hand-crafted rule-based systems to classical machine learning, deep learning and the modern hybrid pipelines that this work builds on is presented in figure 2.1.

2.1 Introduction

There are 5 parts to this chapter. In Section 2.2 we will discuss methods of dealing with class imbalance. In section 2.3 deep-learning techniques that are pertinent to fraud are surveyed. The paper in Section 2.4 explores the application of XAI in financial applications. In Section 2.5 a critical analysis of seven foundational studies is presented that have impacted this work. In Section 2.6 these research gaps are succinctly defined and connected with the contributions defined in Chapter 1.

2.2 Class-Imbalance Handling Techniques

The dominant methodological challenge of fraud detection is the extreme imbalance between fraudulent and legitimate transactions. The ULB benchmark exhibits a ratio near 1:578 [6]; production systems often face ratios closer to 1:5000. A standard classifier minimising a global empirical loss without any imbalance intervention drifts toward the majority class until it becomes operationally useless. Three families of techniques have emerged: data-level resampling, objective-level cost-sensitive learning, and generative methods. Each carries distinct trade-offs. Table 2.1 summarises them.

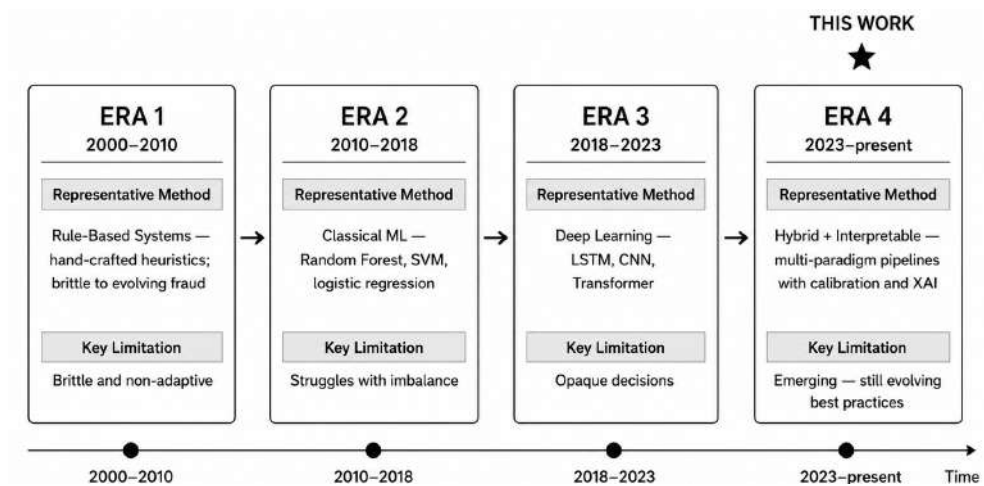


Figure 2.1: Generational evolution of credit-card fraud-detection methodology. The present work sits within the fourth generation, integrating multi-paradigm modelling with formal calibration and three-method explainability.

Table 2.1: Comparative analysis of class-imbalance handling techniques in the fraud-detection literature.

Technique		Strengths	Limitations
Random sampling [37]	under-	Trivial to implement; preserves real distribution	Discards majority information; high variance under extreme imbalance
Random over-sampling [37]	over-	Trivial; full minority retention	Severe overfitting to duplicated examples
SMOTE [16]		Generates synthetic samples via k -NN interpolation	Sensitive to k ; introduces unrealistic decision-boundary patterns
Borderline-SMOTE [17]		Targets the decision boundary specifically	Inherits SMOTE’s artefacts; computationally heavier
ADASYN [38]		Adaptive density-based generation	Can amplify noise in dense minority regions
Class-weighted loss [37, 39]		Preserves the original distribution	Calibration degrades; weight tuning is non-trivial
Threshold tuning [8, 9]		Decouples training from operational threshold	Provides no additional information; needs reliable validation calibration
CTGAN [18]		Models multi-modal joint distribution; tabular-specific design	Sensitive to small training-set size; risk of mode collapse; opaque training dynamics
Hybrid OCSVM-SMOTE-undersampling [40]		Distribution-preserving; explicit pre-and-post comparison	Architectural complexity; multiple hyperparameters

40 The Synthetic Minority Over-sampling Technique [16] is the most widely cited imbalance technique. It generates synthetic minority examples by linearly interpolating between an existing minority point and one of its k -nearest neighbours. Several variants tighten the basic procedure. Borderline-SMOTE [17] targets only points near the decision boundary; ADASYN [38] adapts the number of synthetic samples to the local density of majority points. None of these linear-interpolation methods can reproduce the multimodal joint structure that real fraud distributions exhibit, which has driven the move toward generative methods.

The Conditional Tabular GAN [18] learns a continuous joint distribution from minority training samples and generates new points by sampling from this learned distribution. CTGAN incorporates two design choices specific to tabular data: mode-specific normalisation that handles multimodal continuous columns, and conditional sampling that respects the discrete column structure. This work adopts CTGAN and confines the generated samples to the augmented tabular track only, an arrangement justified in Chapter 4.

Cost-sensitive methods [39] attack the imbalance problem at the objective level rather than the data level. They re-weight per-class contributions to the loss without modifying the training distribution. The trade-off is calibration: re-weighted losses produce probability scores that no longer match empirical class frequencies, requiring post-hoc correction through Platt scaling [19] or isotonic regression [32]. This work introduces a data-level approach to intervention, which is different from the approach used in the author's earlier conference work [1] which was purely threshold tuning.

2.3 Deep Learning for Fraud Detection

Fraud detection has been transformed in two directions by deep learning: sequence modeling and representation learning that works without supervision. This section reviews both.

The first deep architectures used for fraud sequences were recurrent networks. The vanishing gradient problem of vanilla RNNs is mitigated by LSTM units [41], which have input, forget, and output gates to determine the state of the network at each time step. Yousefimehr and Ghatee [40] recently used an LSTM detector with LightGBM gradient-boosted classifier for detecting fraud in sequences and achieved impressive results in the absence of imbalances. A drawback of recurrent networks is the lack of parallelism. They aren't independent, and as a result, don't provide the

within-sequence parallelism that is available on modern hardware.

Transformer encoders [12] eliminate the need for recurrence by instead enabling each position in a sequence to attend to every other position. The key idea of sinusoidal positional encoding is to add sequence ordering as additive features, while maintaining the parallelism lost in recurrent networks. This work proposes a Time-Aware Transformer which extends the input channel by adding the inter-arrival time gaps as a temporal feature, instead of using a positional bias matrix. Convolutional architectures have also been considered. With the aid of SMOTE oversampling, Wu et al. [42] obtain extremely high metrics (accuracy 0.9998, F1 0.9998, recall 1.0000) with Continuous-Coupled Neural Network. Reported performance should be taken with a pinch of salt. Extreme metrics on heavily oversampled training data can reflect distributional artefacts rather than genuine generalisation, and the absence of explainability analysis renders such systems problematic for regulated deployment.

Autoencoders provide unsupervised anomaly detection without requiring fraud labels at training time [28]. The encoder maps inputs into a compressed latent representation; the decoder reconstructs the input from the latent code. Training on legitimate transactions only causes the autoencoder to learn the manifold of normal behaviour. Fraudulent transactions, lying outside this manifold, produce high reconstruction errors that serve as anomaly scores. Fanai and Abbasimehr [29] extend this idea with deep classifiers operating on autoencoder latent representations; Ding et al. [43] couple an autoencoder with LightGBM for combined unsupervised and supervised detection. This work incorporates autoencoder reconstruction error as a feature into both gradient-boosted base learners, an approach validated by the cross-method attribution analysis reported in Chapter 6.

2.4 Explainable Artificial Intelligence in Finance

Two regulatory developments have made explainability a deployment requirement rather than a research preference. The European Union's General Data Protection Regulation establishes a right to meaningful information about automated decisions [36]. The Reserve Bank of India's master directions on digital lending similarly require lenders to explain adverse decisions to affected customers. Both have pushed practitioners toward methods that produce per-decision justifications.

The literature is dominated by two families of explainability methods: the post-hoc methods, which explain the trained black-box models without changing their

structure; and the model-agnostic methods, which are models themselves that can be applied to any black-box model. The basis of attribution in cooperative game theory is SHAP [13] features which quantify the contribution of each feature to the prediction. The TreeSHAP variant takes advantage of the tree structure of gradient boosted models to compute exact Shapley values in polynomial time [33]. LIME [14] samples perturbations of the inputs to fit a locally faithful linear model around each prediction. Both are model agnostic, but neither takes the place of intrinsic interpretability.

Intrinsic methods are methods that are designed to be interpretable by building it into the model. The Explainable Boosting Machine (EBM) [15] is a practical implementation of the Generalised Additive Model with Interactions (GA²M) [15]. The shape functions are not approximations, they are an approximation of the model. This ensures that the visualisation is faithful by construction as opposed to post hoc attributions, where the fidelity is determined by the assumptions of the method for providing the attribution. This paper introduces all three methods together (EBM intrinsic shape functions, TreeSHAP for XGBoost, and LIME for the calibrated ensemble), and examines the feature importance consistency across methods. Only a few papers in the fraud-detection literature report applying more than one explainability method or none of them, at all.

2.5 Critical Analysis of Foundational Studies

Seven recent studies have most directly shaped this work. Their methodological choices and reported metrics are summarised in Table 2.2; this section examines each in critical detail. Pumsirirat and Yan [28] were among the first to combine autoencoders with restricted Boltzmann machines for fraud detection. Their hybrid achieves competitive AUC but does not address class imbalance directly, and reports neither probability calibration nor formal explainability.

Carcillo et al. [44] take a hybrid approach combining unsupervised anomaly scores with supervised classifiers, arguing that the two information sources are complementary. The argument holds, and this work applies the same principle through autoencoder reconstruction error as a feature in supervised base learners.

Saraf and Phakatkar [45] train Random Forest, autoencoder, and Isolation Forest models in parallel and combine them, reporting AUC of 0.9827 on real ULB data. The ensemble construction is principled but the work omits both calibration and explainability.

Table 2.2: Comparative summary of seven foundational fraud-detection studies and the present work. AUC-ROC values are reported on benchmark splits; “Calib?” indicates whether formal probability calibration is reported.

Study	Approach	Imbalance	Key metrics	Calib? / XAI?
Pumsirirat & Yan [28] (2018)	Autoencoder + RBM	None	AUC near 0.96	No / No
Carcillo et al. [44] (2021)	Hybrid supervised + unsupervised	None	F1 0.85 on real data	No / No
Saraf & Phakatkar [45] (2022)	RF + AE + IF hybrid	Real & simulated	AUC 0.9827 (real)	No / No
Fanai & Abbasimehr [29] (2023)	Deep AE + classifier	None	F1 0.91 on ULB	No / No
Khalid et al. [46] (2024)	SVM + KNN + RF ensemble	SMOTE + undersampling	F1 above 0.93	No / Partial
Rathour & Patidar [1] (2025)	RF + AE + IF, threshold-tuned	None	F1 0.88, AUC 0.97	No / No
Wu et al. [42] (2025)	CCNN + SMOTE	SMOTE	F1 0.9998, recall 1.0	No / No
This work	TX + XGB + EBM, calibrated	CTGAN	F1 0.8088, MCC 0.8117, AUC-PR 0.8244	Yes / Yes (3-method)

Fanai and Abbasimehr [29] report strong F1 values on ULB through a deep autoencoder feeding a deep classifier, but again without calibration or attribution analysis.

Khalid et al. [46] combine three classifiers with SMOTE-based augmentation, achieving F1 above 0.93. The work provides partial feature-importance analysis but no formal calibration.

The author’s prior conference paper [1] established the threshold-tuned hybrid framework that this work extends. It deliberately omits oversampling on grounds of distributional integrity, an argument this work revisits through the constrained CTGAN approach described in Chapter 4.

Wu et al. [42] report exceptionally high metrics through SMOTE-augmented Continuous-Coupled Neural Networks. As discussed in Section 2.3, such metrics warrant caution given the absence of validation against distributional artefacts and the

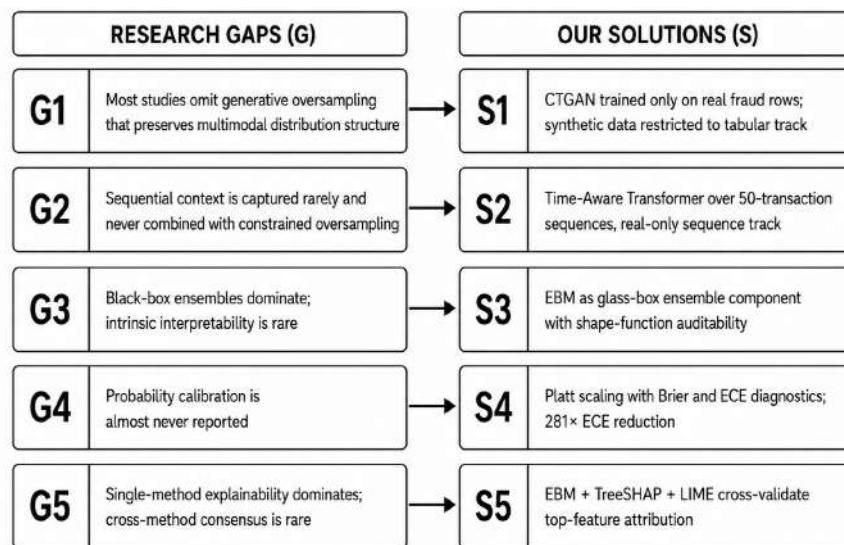


Figure 2.2: Mapping of the five research gaps identified through literature review (left) to the five corresponding solutions introduced in this work (right).

lack of explainability analysis required for regulated deployment.

A consistent pattern emerges across these studies: most achieve strong predictive metrics but address only a subset of the four operational requirements that practical deployment demands. Calibrated probability scores, multi-method explainability, and distribution-preserving oversampling have rarely been combined in the same pipeline.

2.6 Research Gaps and Mapping to work Contributions

Five research gaps follow from the analysis above. Figure 2.2 summarises the gaps and the contributions through which this work addresses each.

Gap G1: Distribution-preserving oversampling on small fraud sets. Most studies either omit oversampling or apply SMOTE-family methods whose linear interpolation cannot reproduce the multimodal joint structure of real fraud distributions. **Solution S1:** CTGAN trained on the 344 real fraud rows of the training partition produces 132 336 synthetic samples confined to the augmented tabular track.

Gap G2: Sequential context combined with constrained oversampling. Sequence models (LSTM, Transformer) are typically trained on imbalanced data without oversampling, while oversampled approaches typically use per-transaction classifiers. **Solution S2:** A Time-Aware Transformer is trained on the real-only sequence track, while the augmented tabular track feeds the gradient-boosted base learners. The two-track architecture preserves both temporal causality and oversampled training depth.

Gap G3: Glass-box components in hybrid ensembles. Existing hybrid ensembles use either fully black-box components (deep networks, gradient-boosted trees) or rely on post-hoc explanation methods. Intrinsically interpretable components are rare. **Solution S3:** The Explainable Boosting Machine [15] is included as a base learner, providing shape functions that constitute the model itself rather than an approximation of it.

Gap G4: Probability calibration. Almost no fraud-detection studies report calibration metrics, despite the operational importance of calibrated probabilities for cost-sensitive thresholding. **Solution S4:** Platt scaling [19] cuts test-set Expected Calibration Error from 0.498 to 0.0018, a $281\times$ improvement, with full Brier score [20] and ECE [21] diagnostics.

Gap G5: Cross-method explainability validation. Most studies report a single explainability method or none. Even when two or three methods are reported, their attributions are rarely compared. **Solution S5:** EBM shape functions [15], TreeSHAP applied to XGBoost [13], and LIME applied to the calibrated ensemble [14] are computed in parallel; their top-feature rankings are compared for consensus.

The remaining chapters develop these five solutions in detail. Chapter 3 provides the theoretical background; Chapter 4 unifies them into a single pipeline; Chapter 5 documents their realisation in code; Chapter 6 reports their empirical performance; Chapter 7 discusses their limitations and identifies future extensions.

2.7 Summary

This chapter has reviewed the literature on credit-card fraud detection across four methodological dimensions and identified the research gaps the present work addresses. Section 2.2 covered class-imbalance handling techniques, contrasting data-level resampling methods such as SMOTE [16] and its variants [17, 38] with objective-level cost-sensitive learning [39] and modern generative methods including CTGAN [18]. The review concluded that linear-interpolation methods cannot reproduce the multi-modal joint structure of real fraud distributions, motivating the adoption of CTGAN in the present work.

Section 2.3 examined deep-learning approaches relevant to fraud detection, covering LSTM-based [40, 41] sequential models, Transformer encoders [12], convolutional architectures including the recent CCNN work [42], and autoencoder-based unsupervised anomaly detection [28, 29, 43]. The review identified the absence of sequential

modelling combined with constrained oversampling as a persistent gap in the literature.

Section 2.4 treated explainable artificial intelligence in financial contexts, distinguishing post-hoc methods such as SHAP [13] and LIME [14] from intrinsic methods such as the Generalised Additive Model with Interactions [15, 31]. The regulatory pressure from the EU GDPR right to explanation [36] and similar Indian directives makes interpretability a deployment requirement rather than a research preference, yet most fraud-detection studies report only one explainability method or none at all.

Section 2.5 provided a critical analysis of seven foundational studies that have shaped the present work. Most of these studies report strong predictive metrics on the ULB benchmark [6, 7] but address only a subset of the four operational requirements that practical deployment demands. Calibrated probability scores, multi-method explainability, and distribution-preserving oversampling have rarely been combined in a single pipeline.

Section 2.6 articulated five specific research gaps and mapped each to a corresponding contribution of the present work: distribution-preserving oversampling on small fraud sets ($G1 \rightarrow S1$); sequential context combined with constrained oversampling ($G2 \rightarrow S2$); glass-box components in hybrid ensembles ($G3 \rightarrow S3$); probability calibration ($G4 \rightarrow S4$); and cross-method explainability validation ($G5 \rightarrow S5$). The remaining chapters develop these five solutions in detail. Chapter 3 provides the theoretical background; Chapter 4 unifies the solutions into a single pipeline; Chapter 5 documents their realisation in code; Chapter 6 reports their empirical performance; and Chapter 7 discusses the limitations and identifies future extensions.

CHAPTER 3

THEORETICAL BACKGROUND

The methodology developed in Chapter 4 draws on seven theoretical pillars, each from a distinct research tradition. Figure 3.1 groups them into the four functional roles they play in the proposed pipeline: generative augmentation, representation learning, supervised classification, and post-processing.

3.1 Introduction

The chapter presents each pillar formally. Section 3.2 develops the Conditional Tabular GAN with the WGAN-GP training objective. Section 3.3 treats the autoencoder and reconstruction-error anomaly scoring. Section 3.4 formalises self-attention and the Transformer encoder. Section 3.5 develops gradient boosting with the Explainable Boosting Machine as a glass-box variant. Section 3.6 introduces Platt scaling with Brier score and Expected Calibration Error as diagnostics. Section 3.7 treats the Shapley value framework underlying SHAP together with the local-linear approximation underlying LIME.

3.2 Conditional Tabular GAN

A generative adversarial network [26] consists of a generator G_θ that maps latent samples $z \sim p_z$ to synthetic data points and a discriminator D_ϕ that distinguishes real from synthetic samples. The original GAN objective minimises the Jensen-Shannon divergence between the real and generated distributions. Training with this objective is notoriously unstable.

The Wasserstein GAN with gradient penalty (WGAN-GP) [27] replaces the Jensen-Shannon-based discriminator with a critic that estimates the Wasserstein-1 distance between distributions, with the critic loss given in Equation 3.1:

$$\mathcal{L}_D = \mathbb{E}_{\tilde{x} \sim p_g} [D_\phi(\tilde{x})] - \mathbb{E}_{x \sim p_r} [D_\phi(x)] + \lambda \mathbb{E}_{\hat{x} \sim p_{\hat{x}}} [(\|\nabla_{\hat{x}} D_\phi(\hat{x})\|_2 - 1)^2], \quad (3.1)$$

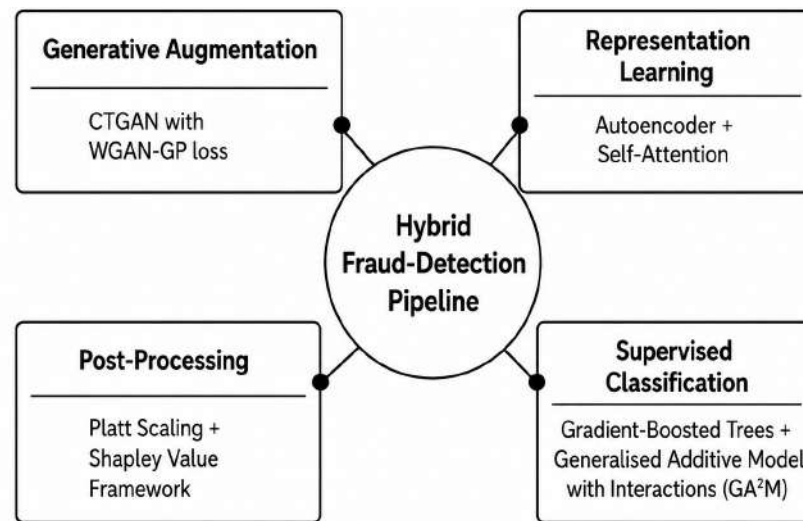


Figure 3.1: The seven theoretical pillars supporting the proposed methodology, grouped by their functional role in the pipeline

where p_g is the generator distribution, p_r is the real distribution, $\hat{x}$ is sampled along straight lines between real and generated points, and λ controls the gradient penalty strength. The gradient penalty term in Equation 3.1 enforces the 1-Lipschitz constraint required by the Wasserstein formulation without the weight clipping that destabilises the original WGAN.

The Conditional Tabular GAN [18] adapts WGAN-GP to tabular data through three design choices. **Mode-specific normalisation** handles multimodal continuous columns by fitting a Gaussian mixture per column and representing each value as a one-hot mode indicator plus a within-mode standardised value. **Conditional sampling** selects discrete-column values from a uniform distribution during training so that the generator learns the full joint distribution rather than the empirical marginal. The **training-by-sampling** procedure ensures both rare and common discrete values are seen during training.

For fraud detection, CTGAN's appeal is its capacity to capture multimodal joint structure in the PCA-transformed ULB feature space, which the linear interpolation of SMOTE [16] cannot reproduce.

3.3 Autoencoder and Reconstruction-Error Anomaly Scoring

An autoencoder consists of an encoder $f_\theta : \mathbb{R}^d \rightarrow \mathbb{R}^k$ and a decoder $g_\phi : \mathbb{R}^k \rightarrow \mathbb{R}^d$ trained jointly to minimise the reconstruction error defined in Equation 3.2:

$$\mathcal{L}_{\text{AE}}(\theta, \phi) = \mathbb{E}_{x \sim p_{\text{legit}}} [\|x - g_\phi(f_\theta(x))\|_2^2]. \quad (3.2)$$

Training under the objective in Equation 3.2 on legitimate transactions alone causes the encoder to learn a compressed representation of the normal-behaviour manifold. Fraudulent transactions, lying outside this manifold, produce high reconstruction errors as defined in Equation 3.3:

$$\varepsilon(x) = \|x - g_\phi(f_\theta(x))\|_2^2, \quad (3.3)$$

which serve as anomaly scores. The score in Equation 3.3 is identical in form to the per-sample term inside the expectation of Equation 3.2, but is evaluated at inference time on individual transactions. The compression dimension k trades off representation fidelity against discriminative power. Small k forces the network to learn only the most salient features of legitimate transactions, sharpening the ε signal for fraud.

Autoencoder-based fraud detection was first applied by Pumsirirat and Yan [28] and has since been refined by Fanai and Abbasimehr [29] and others. This work builds an autoencoder with a symmetric architecture $d \rightarrow 128 \rightarrow 64 \rightarrow 32 \rightarrow 64 \rightarrow 128 \rightarrow d$, treating ε as a feature input to the gradient-boosted base learners rather than as a standalone classifier.

3.4 Self-Attention and the Transformer Encoder

The Transformer encoder [12] replaces recurrence with self-attention. Given an input sequence $X \in \mathbb{R}^{n \times d_{\text{model}}}$ of n transactions, three learned linear projections produce queries Q , keys K , and values V , as defined in Equation 3.4:

$$Q = XW_Q, \quad K = XW_K, \quad V = XW_V, \quad (3.4)$$

with $W_Q, W_K \in \mathbb{R}^{d_{\text{model}} \times d_k}$ and $W_V \in \mathbb{R}^{d_{\text{model}} \times d_v}$. Using the projections from Equation 3.4, scaled dot-product attention is computed as shown in Equation 3.5:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^\top}{\sqrt{d_k}}\right)V. \quad (3.5)$$

The scaling factor $\sqrt{d_k}$ in Equation 3.5 keeps the softmax from saturating when d_k is large.

Multi-head attention concatenates h parallel attention heads following the construction in Equation 3.6:

$$\text{MultiHead}(X) = \text{Concat}(\text{head}_1, \dots, \text{head}_h)W_O, \quad \text{head}_i = \text{Attention}(XW_Q^{(i)}, XW_K^{(i)}, XW_V^{(i)}). \quad (3.6)$$

As shown in Equation 3.6, each head learns to attend to a different aspect of the input sequence; their outputs are combined through the projection matrix $W_O \in \mathbb{R}^{hd_v \times d_{\text{model}}}$.

A Transformer encoder layer combines multi-head attention with a position-wise feed-forward network and applies residual connections plus layer normalisation. Self-attention has no intrinsic notion of sequence order, so positional encoding adds a position-dependent vector to each input. This work uses the sinusoidal positional encoding given in Equation 3.7:

$$\text{PE}_{(p,2i)} = \sin\left(\frac{p}{10000^{2i/d_{\text{model}}}}\right), \quad \text{PE}_{(p,2i+1)} = \cos\left(\frac{p}{10000^{2i/d_{\text{model}}}}\right), \quad (3.7)$$

where in Equation 3.7, p is the position index and i the dimension index.

Inter-arrival time information enters as an additional input channel rather than as a positional bias matrix. This approach keeps the standard Transformer architecture intact while supplying temporal structure as a feature.

3.5 Gradient Boosting and the Explainable Boosting Machine

Gradient boosting [30] builds an additive model $F_M(x) = \sum_{m=1}^M \eta h_m(x)$ sequentially. Each weak learner h_m fits the negative gradient of a loss function ℓ evaluated at the current ensemble's predictions, as expressed in Equation 3.8:

$$h_m \approx \arg \min_h \sum_i \left(-\nabla_{F_{m-1}(x_i)} \ell(y_i, F_{m-1}(x_i)) - h(x_i) \right)^2, \quad (3.8)$$

where $\eta \in (0, 1]$ is the learning rate that controls the contribution of each successive learner. The fitting rule in Equation 3.8 is the foundation on which XGBoost [23] extends gradient boosting with second-order Taylor expansion, regularised leaf weights, and sparsity-aware splitting, and dominates contemporary tabular benchmarks.

The Generalised Additive Model with Interactions (GA²M) [15, 31] constrains

gradient boosting to produce an interpretable additive structure, as shown in Equation 3.9:

$$g(\mathbb{E}[y | x]) = \beta_0 + \sum_j f_j(x_j) + \sum_{(j,k) \in \mathcal{J}} f_{j,k}(x_j, x_k), \quad (3.9)$$

where, in Equation 3.9, g is the link function (sigmoid for binary classification), f_j are per-feature shape functions, and $f_{j,k}$ are pairwise interaction shape functions for a deliberately small set $\mathcal{J}$. The Explainable Boosting Machine [15] fits the GA^2M form in Equation 3.9 through cyclic coordinate-descent gradient boosting, producing shape functions that are themselves the model rather than a post-hoc approximation.

The interpretability advantage is structural. Any individual prediction can be decomposed exactly into per-feature contributions, and any feature's effect can be visualised as a plot of $f_j(x_j)$ over the feature's range. Predictive performance is typically within one to two AUC points of full gradient-boosted trees on tabular benchmarks.

3.6 Probability Calibration

A classifier producing scores $s(x)$ is *calibrated* if $\Pr(y = 1 | s(x) = p) = p$ for every $p \in [0, 1]$. Modern classifiers are typically miscalibrated even when their ranking performance is strong; rank-averaged ensembles in particular produce uniform $(0, 1]$ scores that bear no relationship to empirical class probabilities.

Platt scaling [19] maps raw scores to calibrated probabilities through a univariate logistic regression, as defined in Equation 3.10:

$$\hat{p}_{\text{cal}}(x) = \sigma(A \cdot s(x) + B), \quad (3.10)$$

where in Equation 3.10, σ is the logistic function and the parameters (A, B) are fitted by maximum likelihood on a held-out validation set. The validation set must not have been used during model training, and the test set must not be used during calibration fitting.

Two diagnostics quantify calibration quality. The Brier score [20] is the mean squared error between predicted probability and binary outcome, as given in Equation 3.11:

$$\text{Brier} = \frac{1}{N} \sum_{i=1}^N (\hat{p}(x_i) - y_i)^2. \quad (3.11)$$

The Expected Calibration Error [21] partitions predictions into B equal-width probability bins and measures the gap between mean predicted probability and empirical positive

rate within each bin, as defined in Equation 3.12:

$$\text{ECE} = \sum_{b=1}^B \frac{|S_b|}{N} |\text{conf}(S_b) - \text{acc}(S_b)|, \quad (3.12)$$

where, in Equation 3.12, S_b is the set of predictions falling in bin b , $\text{conf}(S_b)$ is their mean predicted probability, and $\text{acc}(S_b)$ is the empirical positive rate among them. A perfectly calibrated classifier achieves $\text{ECE} = 0$ under Equation 3.12.

Platt scaling's two-parameter form (Equation 3.10) suits small positive-class counts well. The more flexible isotonic regression [32] risks overfitting under the same conditions.

3.7 Shapley Values and Local-Linear Approximation

Cooperative game theory gives rise to a framework for feature attribution which is uniquely axiomatic, namely the Shapley value [33]. The Shapley value of feature j is the average of its marginal contribution with respect to all subsets of features with respect to model f and instance x as shown in Equation 3.13:

$$\phi_j(x) = \sum_{S \subseteq F \setminus \{j\}} \frac{|S|! (|F| - |S| - 1)!}{|F|!} [f_{S \cup \{j\}}(x) - f_S(x)], \quad (3.13)$$

In Equation 3.13 F denotes the entire feature set and f_S the model that ignores the features in S . The Shapley value (Equation 3.13) is the only attribution method that is efficient, symmetric, dummy and additive.

SHAP [13] is a variant of Shapley values in machine learning. The TreeSHAP variant exploits the structure of gradient-boosted trees to compute exact Shapley values in polynomial time, a substantial improvement over the exponential cost of the brute-force formula in Equation 3.13. For an XGBoost model, TreeSHAP enables per-instance attribution at scale.

LIME [14] takes a different approach. Rather than computing exact game-theoretic attributions, it fits a locally faithful linear model around each prediction. For an instance x , LIME samples nearby points z' , weights them by a proximity kernel $\pi_x(z')$, and minimises the objective in Equation 3.14:

$$\xi(x) = \arg \min_{g \in G} \mathcal{L}(f, g, \pi_x) + \Omega(g), \quad (3.14)$$

where, in Equation 3.14, G is a class of interpretable models (typically sparse linear), $\mathcal{L}$ is a fidelity loss, and Ω penalises explanation complexity. LIME's locality assumption is its weakness: an explanation faithful around x may not generalise to nearby points.

This work applies all three approaches in combination — EBM intrinsic shape functions, TreeSHAP applied to the XGBoost component, and LIME applied to the calibrated full ensemble — and validates feature importance through cross-method consensus. A feature consistently ranked high by three structurally different methods is unlikely to be a method-specific artefact.

3.8 Summary

This chapter has presented the seven theoretical pillars underlying the methodology of Chapter 4. Section 3.2 formalised the Conditional Tabular GAN with the WGAN-GP training objective [18, 27], which produces synthetic fraud samples that preserve multimodal joint structure unattainable through linear interpolation methods such as SMOTE [16]. Section 3.3 developed reconstruction-error anomaly scoring, in which a symmetric autoencoder trained only on legitimate transactions produces an anomaly signal $\varepsilon(x)$ that feeds directly into the gradient-boosted base learners as the `ae_error` feature. Section 3.4 formalised the Transformer encoder [12], including scaled dot-product attention, multi-head extension, and sinusoidal positional encoding. Section 3.5 developed gradient boosting [23, 30] together with the Generalised Additive Model with Interactions that the Explainable Boosting Machine [15] implements as a glass-box alternative. Section 3.6 presented Platt scaling [19] and the Brier score [20] and Expected Calibration Error [21] diagnostics. Section 3.7 treated the Shapley value [33] framework underlying TreeSHAP [13] together with the local-linear approximation underlying [14].

These seven theoretical pillars are not deployed in isolation. The pipeline of Chapter 4 unifies them into a single end-to-end procedure that progresses from raw transaction records through generative augmentation, dual-track feature engineering, hybrid ensemble training, calibration, threshold tuning, and three-method explainability cross-validation.

CHAPTER 4

PROPOSED METHODOLOGY

4.1 Overview of the Proposed Framework

The proposed framework integrates the seven theoretical pillars of Chapter 3 into a single end-to-end pipeline that takes raw transaction records as input and produces calibrated fraud probabilities, threshold-tuned classifications, and auditable per-feature attributions as output. The entire process is provided below in Algorithm 1. It defines the whole process—from raw inputs in transactions to calibrated probabilities to threshold-tuned classifications to three-method attribution—all in one. The pipeline is not modular, but rather the artefacts created by each step are fed to the next step, and the system is trained and evaluated as a single unit.

Algorithm 1 is read as a top-down trace of the working system. The first block (lines 1–8) prepares the data: the Hour feature is derived from raw time, the two-stage scaler pair is selected through a Taguchi L_{25} [24] cross-validated search, and the resulting transformation is applied before a stratified 70/15/15 partition. The second block (lines 9–12) handles class imbalance and prepares the dual data view: a CTGAN [18] is fitted on the real fraud rows, the synthetic sample count is derived from the target 40 % augmented fraud ratio (Equation 4.1), and the augmented tabular track $\mathcal{D}_{\text{tab}}$ is separated from the real-only sequence track $\mathcal{D}_{\text{seq}}$. The third block (lines 13–16) performs feature engineering: ten temporal features and the cyclic Hour encoding (Equation 4.3) are computed, and an autoencoder pre-trained on legitimate-only data contributes a reconstruction-error feature and a 32-dimensional latent embedding to both tracks. The fourth block (lines 17–19) trains the three structurally diverse base learners: a Time-Aware Transformer [12] on the sequence track, and XGBoost [23] and an EBM [15] on the augmented tabular track. The fifth block (lines 20–23) aggregates the base learners through rank averaging (Equation 4.5), calibrates the ensemble through Platt scaling (Equation 4.6) [19], and tunes operational thresholds on the validation set. The last block (line 24) creates three separate attributions and validates the three in order to find a consensus. The algorithm ends with returning the calibrated probability function and the threshold set and the three-method attribution.

Algorithm 1 Hybrid Interpretable Credit-Card Fraud Detection Pipeline.

Input: Raw transaction stream $\mathcal{T} = \{(x_i, y_i)\}_{i=1}^N, x_i \in \mathbb{R}^d, y_i \in \{0, 1\}$

Output: Calibrated fraud probabilities $\hat{p}_{\text{cal}}(\cdot)$, threshold τ^* , three-method attribution

- 1: Extract Hour $\leftarrow \lfloor \text{Time}/3600 \rfloor \bmod 24$; drop Time from features
- 2: Run Taguchi $L_{25}(5 \times 5)$ orthogonal-array search:
- 3: **for** each pair $(S_1, S_2) \in \{\text{MinMax, Standard, Quantile, Robust, Power}\}^2$ **do**
- 4: Evaluate via stratified 5-fold CV on $\mathcal{D}_{\text{train}}$ with logistic regression as base classifier
- 5: Refit scalers within each fold to prevent leakage
- 6: **end for**
- 7: $(S_1^*, S_2^*) \leftarrow \arg \max$ mean cross-validated AUC-ROC
- 8: Apply $S_1^* \rightarrow S_2^*$ to all data; stratified 70/15/15 split into $\mathcal{D}_{\text{train}}, \mathcal{D}_{\text{val}}, \mathcal{D}_{\text{test}}$
- 9: Train CTGAN with WGAN-GP loss on the 344 real fraud rows of $\mathcal{D}_{\text{train}}$ for 300 epochs
- 10: Generate $S = \frac{0.40(L+F)-F}{1-0.40} = 132\,336$ synthetic fraud rows
- 11: $\mathcal{D}_{\text{tab}} \leftarrow \text{real } \mathcal{D}_{\text{train}} \cup \text{synthetic}$ (331 700 rows, used for AE / XGB / EBM)
- 12: $\mathcal{D}_{\text{seq}} \leftarrow \text{real } \mathcal{D}_{\text{train}}$ sorted chronologically (199 364 rows, used for Transformer)
- 13: Engineer 10 temporal features and cyclic Hour encoding $(\sin(2\pi h/24), \cos(2\pi h/24))$ for both tracks; assign zeros to synthetic rows
- 14: Train autoencoder $f_\theta \circ g_\phi$ ($d \rightarrow 128 \rightarrow 64 \rightarrow 32 \rightarrow 64 \rightarrow 128 \rightarrow d$) on legitimate-only rows
- 15: Compute reconstruction error $\mathcal{E}(x_i) = \|x_i - g_\phi(f_\theta(x_i))\|_2^2$ and latent $z_i = f_\theta(x_i)$ for all rows
- 16: Append ae_error and 32-dim latent z to feature matrices of both tracks (final dimension: 73)
- 17: Train Time-Aware Transformer on $\mathcal{D}_{\text{seq}}$ with sliding window $n = 50$, mixed precision, AdamW
- 18: Train XGBoost on $\mathcal{D}_{\text{tab}}$ excluding leakage-prone temporal features (64-feature subset)
- 19: Train Explainable Boosting Machine on $\mathcal{D}_{\text{tab}}$ with class-weighted loss
- 20: **for** each base learner $k \in \{\text{TX, XGB, EBM}\}$ **do**
- 21: Convert probability vector to ranks: $r_i^{(k)} = \text{rank}(p_i^{(k)})/N$
- 22: **end for**
- 23: $\hat{p}_i^{\text{ensemble}} \leftarrow \frac{1}{3}(r_i^{(\text{TX})} + r_i^{(\text{XGB})} + r_i^{(\text{EBM})})$
- 24: Fit Platt scaler on $\mathcal{D}_{\text{val}}$ and compute $\hat{p}_{\text{cal}}(x) = \sigma(A\hat{p}^{\text{ensemble}}(x) + B)$
- 25: Tune thresholds: $\tau_{F1}^*, \tau_{\text{PR}}^*, \tau_{R \geq 0.9}^*$ on $\mathcal{D}_{\text{val}}$
- 26: Compute EBM, TreeSHAP, and LIME attributions; cross-validate top features through three-way consensus
- 27: **return** $\hat{p}_{\text{cal}}(\cdot)$, threshold set $\{\tau_{F1}^*, \tau_{R \geq 0.9}^*\}$, three-method attribution

The two-track data architecture, which splits synthetic-augmented tabular data from real-only sequence data; the rank-averaged ensemble, which aggregates structurally diverse base learners; and the three-method explainability cross-validation, which mitigates artefacts caused by the explainability method, are three design decisions that make this framework distinct from previous research.

4.2 Data Pre-processing

4.2.1 ULB Dataset

The ULB Credit Card Fraud benchmark [6, 7] contains 284 807 European cardholder transactions over a two-day period, of which 492 are labelled fraudulent (class ratio approximately 1:578). Each transaction has 30 features: 28 PCA-transformed components V_1 through V_{28} , Time (seconds since first transaction), and Amount. Initial validation confirms zero missing values, no infinite entries, and no obvious duplicate rows.

4.2.2 Temporal Feature Engineering

The raw Time feature encodes seconds since the first transaction, taking values from 0 to approximately 172,800 (48 hours). 2 transformations follow.

First, the hour of the day is extracted, which is the value of $\lfloor \text{Time}/3600 \rfloor \bmod 24$; this is used to represent diurnal patterns in the occurrence of fraud.

Second, the raw Time column is dropped from the modelling feature set. Retaining a row-unique counter would let tree-based models memorise the training set perfectly without learning any generalisable fraud signal. The raw timestamps are preserved separately for downstream inter-arrival-time computation.

4.2.3 Taguchi L_{25} Scaler Search

Two-stage feature scaling improves classifier convergence on the heterogeneously distributed PCA components. A Taguchi L_{25} orthogonal array [24] searches the joint space of (S_1, S_2) scaler pairs over five candidates each: MinMax, Standard, Quantile, Robust, and Power (Yeo-Johnson). The full $5 \times 5 = 25$ design with two factors is itself the L_{25} array — every level appears five times in each position and every pair exactly once.

To avoid leakage, the scalers are refit *within each cross-validation fold*. The chosen pair on the ULB benchmark is Quantile $\rightarrow$ Robust with mean cross-validated

AUC of 0.97940 ± 0.00564 . The top four configurations are in close agreement, within 0.0003 AUC, which is within the cross-fold standard deviation, and the choice is therefore empirically sound, but not statistically different from other configurations that are also good.

4.2.4 Train-Validation-Test Split

The dataset is partitioned through stratified random sampling in a 70/15/15 ratio. The result appears in Table 4.1.

Table 4.1: Stratified 70/15/15 split of the ULB dataset.

Partition	Total rows	Fraud rows	Fraud rate
Train	199 364	344	0.001725
Validation	42 721	74	0.001732
Test	42 722	74	0.001732
Total	284 807	492	0.001727

The fraud rate is preserved across partitions to within 3×10^{-6} . Chronological splitting is not used because the ULB dataset spans only 48 hours — a window over which fraud patterns do not drift sufficiently to justify time-ordered evaluation.

4.3 Class-Imbalance Handling

4.3.1 CTGAN Configuration

A Conditional Tabular GAN [18] is trained on the 344 real fraud rows of the training partition. Hyperparameters appear in Table 4.2. The Class column is excluded from the CTGAN input (constant on fraud-only data) and re-attached to generated samples.

Table 4.2: CTGAN training configuration.

Hyperparameter	Value
Training epochs	300
Batch size	500
Learning rate (generator & critic)	2×10^{-4}
Generator architecture	2 residual blocks (256, 256)
Critic architecture	2 residual blocks (256, 256)
Optimiser	Adam [47]
Gradient-penalty coefficient λ	10 (WGAN-GP)
Random seed	42
Wall-time on Tesla T4 GPU	28.1 s

4.3.2 Synthetic Sample Size

The target augmented fraud ratio is set at 40 %. Solving $(F + S)/(L + F + S) = 0.40$ for the synthetic-sample count S with $L = 199\,020$ legitimate rows and $F = 344$ real fraud rows yields the closed-form expression given in Equation 4.1:

$$S = \frac{0.40(L + F) - F}{1 - 0.40} = 132\,336. \quad (4.1)$$

As shown in Equation 4.1, the augmented training set thus contains $L + F + S = 331\,700$ rows.

4.3.3 Class-Weighted Loss

After augmentation, class weights for downstream losses are computed as defined in Equation 4.2:

$$w_{\text{fraud}} = \frac{N_{\text{total}}}{2N_{\text{fraud}}} = 1.2500, \quad w_{\text{legit}} = \frac{N_{\text{total}}}{2N_{\text{legit}}} = 0.8333. \quad (4.2)$$

For PyTorch BCEWithLogitsLoss, the equivalent pos_weight is $N_{\text{legit}}/N_{\text{fraud}} = 1.5000$, which also serves as XGBoost's scale_pos_weight. The values reported in Equation 4.2 are propagated unchanged to every downstream class-weighted training objective.

4.4 Two-Track Architecture and Feature Engineering

4.4.1 The Two-Track Construction

Synthetic CTGAN samples have no real timestamps, only generated ones. Including these samples in a Transformer’s input sequences would inject distributional artefacts into the temporal modelling track, where causality matters. The proposed solution is a two-track architecture, illustrated in Figure 4.1.

The tabular track is the union of all real and synthetic rows; the sequence track contains only real transactions, sorted chronologically. The two-track design preserves causal time semantics in the sequence track while permitting CTGAN-augmented training of the tabular learners. Both tracks share the same 73-feature column space (30 base features + 10 temporal features + 1 ae_error + 32 latent z).

4.4.2 Engineered Temporal Features

Ten temporal features are computed on real transactions only: Hour (integer 0–23), inter-arrival time Δt , $\log(1 + \Delta t)$, four Δt -bin indicators (under one hour, 1–24 hours, 1–7 days, over 7 days), 1-hour rolling transaction count, and 24-hour rolling mean and standard deviation of Amount. Synthetic CTGAN rows are assigned zero values for these features — a design choice that makes the temporal features unsuitable for tree-based models trained on $\mathcal{D}_{\text{tab}}$, as discussed in Section 4.5.

4.4.3 Cyclical Hour Encoding

The upstream Quantile → Robust scaling pipeline destroys Hour’s cyclical adjacency (hour 23 and hour 0 should be adjacent but become the extreme values of the scaled distribution). Cyclical re-encoding restores adjacency, as defined in Equation 4.3:

$$\text{Hour}_{\sin} = \sin\left(\frac{2\pi h}{24}\right), \quad \text{Hour}_{\cos} = \cos\left(\frac{2\pi h}{24}\right). \quad (4.3)$$

For the encoding in Equation 4.3, the unit-circle property $\text{Hour}_{\sin}^2 + \text{Hour}_{\cos}^2 = 1$ holds to machine precision; empirical means are approximately zero with standard deviations near $1/\sqrt{2}$, confirming uniform coverage of the 24-hour cycle.

4.4.4 Autoencoder Pre-training

The autoencoder follows the symmetric architecture $d \rightarrow 128 \rightarrow 64 \rightarrow 32 \rightarrow 64 \rightarrow 128 \rightarrow d$ with each fully-connected layer composed of Linear → BatchNorm → ReLU

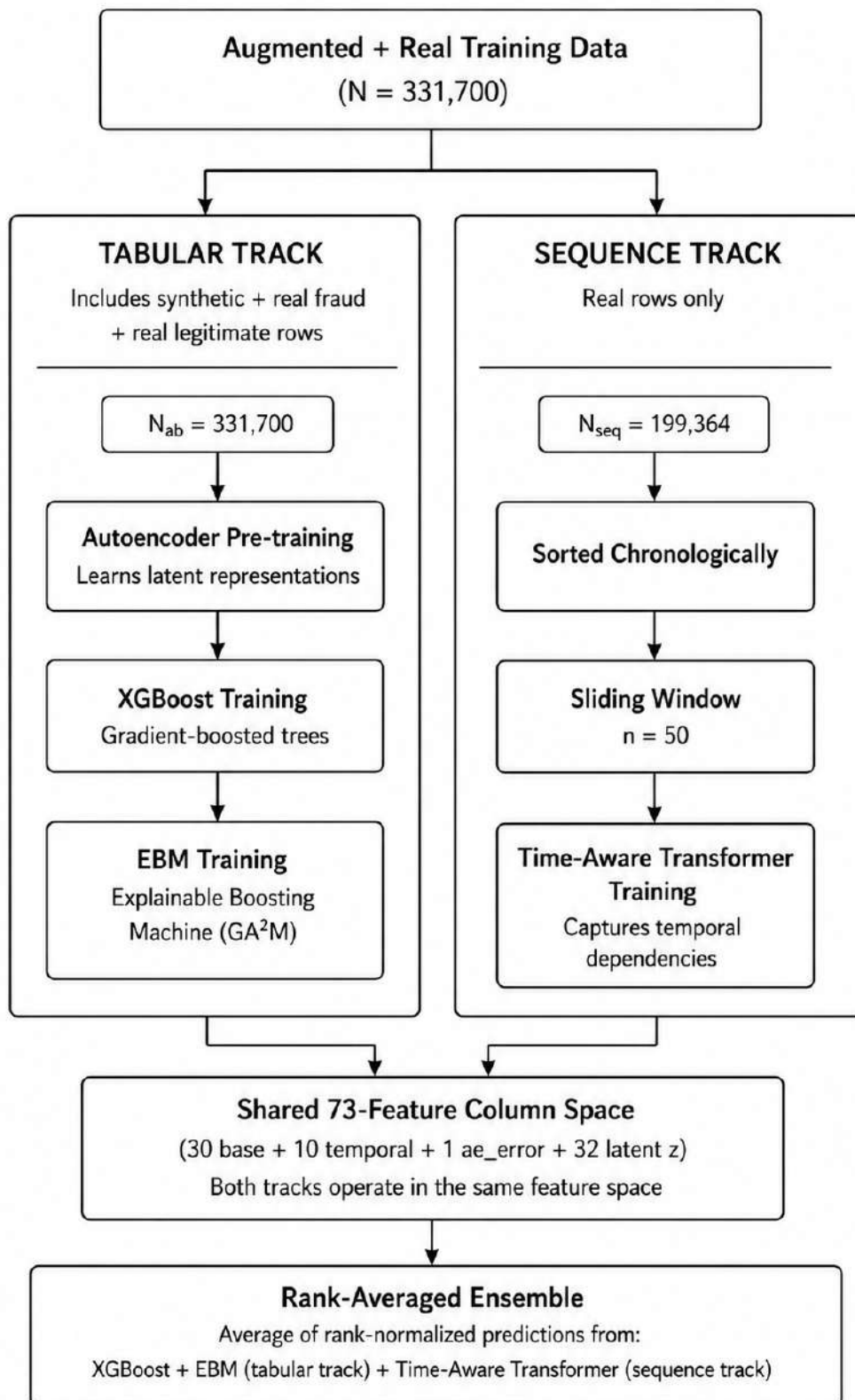


Figure 4.1: Two-track data architecture. Synthetic samples augment the tabular track only; the sequence track preserves real chronological structure required by the Transformer.

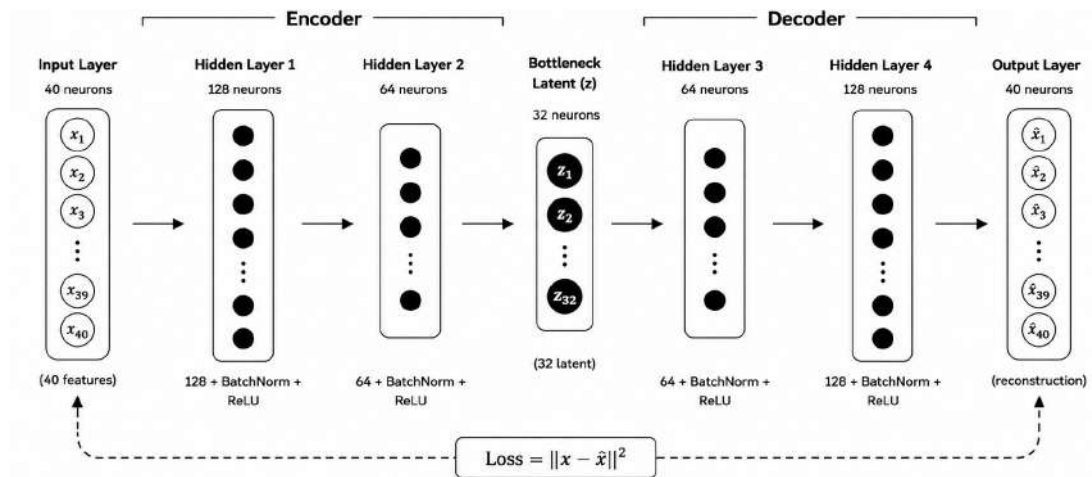


Figure 4.2: Symmetric autoencoder with 32-dimensional bottleneck. The reconstruction error $\varepsilon = \|x - g_\phi(f_\theta(x))\|_2^2$ serves as an anomaly feature for the gradient-boosted base learners.

(the final decoder layer is linear-only). Training uses Adam [47] with learning rate 10^{-3} , batch size 256, on the 199020 real legitimate transactions of the training partition. A held-out 10 % legitimate validation slice triggers early stopping with patience 5.

The trained autoencoder produces a $64\times$ median separation between legitimate-class and fraud-class reconstruction errors on the held-out test set — a strong unsupervised anomaly signal that feeds the gradient-boosted base learners as the `ae_error` feature.

4.4.5 Sequence Construction

For each real transaction i , a window of length $n = 50$ is constructed from rows $\{i - 49, i - 48, \dots, i\}$ in chronological order, with left zero-padding when fewer than 50 predecessors are available. Sequences are constructed lazily (per-batch) to avoid the 2.9 GB memory footprint of an eager representation. Each window is paired with its Δt vector.

4.5 Hybrid Base Learners

The proposed framework is a deliberate reduction of an ambitious five-component proposal. The reductions appear in Table 4.3.

Table 4.3: Reductions from the original specification.

Original specification	Implemented version	Justification
Custom temporal-bias attention $B(\Delta t_{ij})$	Δt as 74 th input channel	Standard attention learns Δt -aware patterns from input.
Recency-weighted pooling	Last-token pooling	Identical to recency weighting at $\gamma \rightarrow 1$; simpler.
~ 6 M-param Transformer (4 layers, 8 heads, $d = 256$)	282,881-param Transformer (2 layers, 4 heads, $d = 128$)	Larger model overfits with only 344 fraud examples.
CNN with SE attention	Dropped	Gradient-boosted trees outperform 1-D CNN on tabular data.
4 base learners + LogReg meta-learner stacking	3 base learners + rank averaging	Rank averaging avoids OOF leakage and matches stacking on small data [25].

4.5.1 Time-Aware Transformer

The Transformer encoder concatenates Δt as an additional input channel, projects via a linear layer to $d_{\text{model}} = 128$, applies sinusoidal positional encoding [12], passes through two pre-LayerNorm encoder layers with 4 attention heads each and FFN dimension 256 (GELU activation), then routes the last-token output through a small classification head. The full forward pass is summarised in Equation 4.4:

$$x'_i = \text{Linear}_{74 \rightarrow 128}([x_i \mid \Delta t_i]) + \text{PE}(i), \quad h = \text{TransformerEncoder}(x'), \quad \hat{p}_{\text{TX}} = \sigma(\text{MLP}(h_{[-1]})). \quad (4.4)$$

As shown in Equation 4.4, the time channel enters the model at the input projection rather than as a custom attention bias, in line with the reduction listed in Table 4.3.

Training uses AdamW (learning rate 10^{-4} , weight decay 10^{-5}), cosine annealing with warm restarts, weighted BCEWithLogitsLoss (pos_weight = 1.5), mixed-precision [34] with gradient clipping at max-norm 1.0, batch size 32, 20 maximum epochs, and early stopping on validation AUC-ROC. The best checkpoint is reached at epoch 8 with total wall-time of 21.2 minutes on Tesla T4.

4.5.2 XGBoost with Leakage Mitigation

An initial XGBoost [23] fit on all 73 features yielded test AUC-PR = 0.83748 with feature `dt_1t_1h` dominating at importance 0.58 — a methodological problem. The synthetic CTGAN rows have zero-valued temporal features (assigned in Section 4.4), so XGBoost was implicitly using these features as a synthetic-vs-real indicator rather

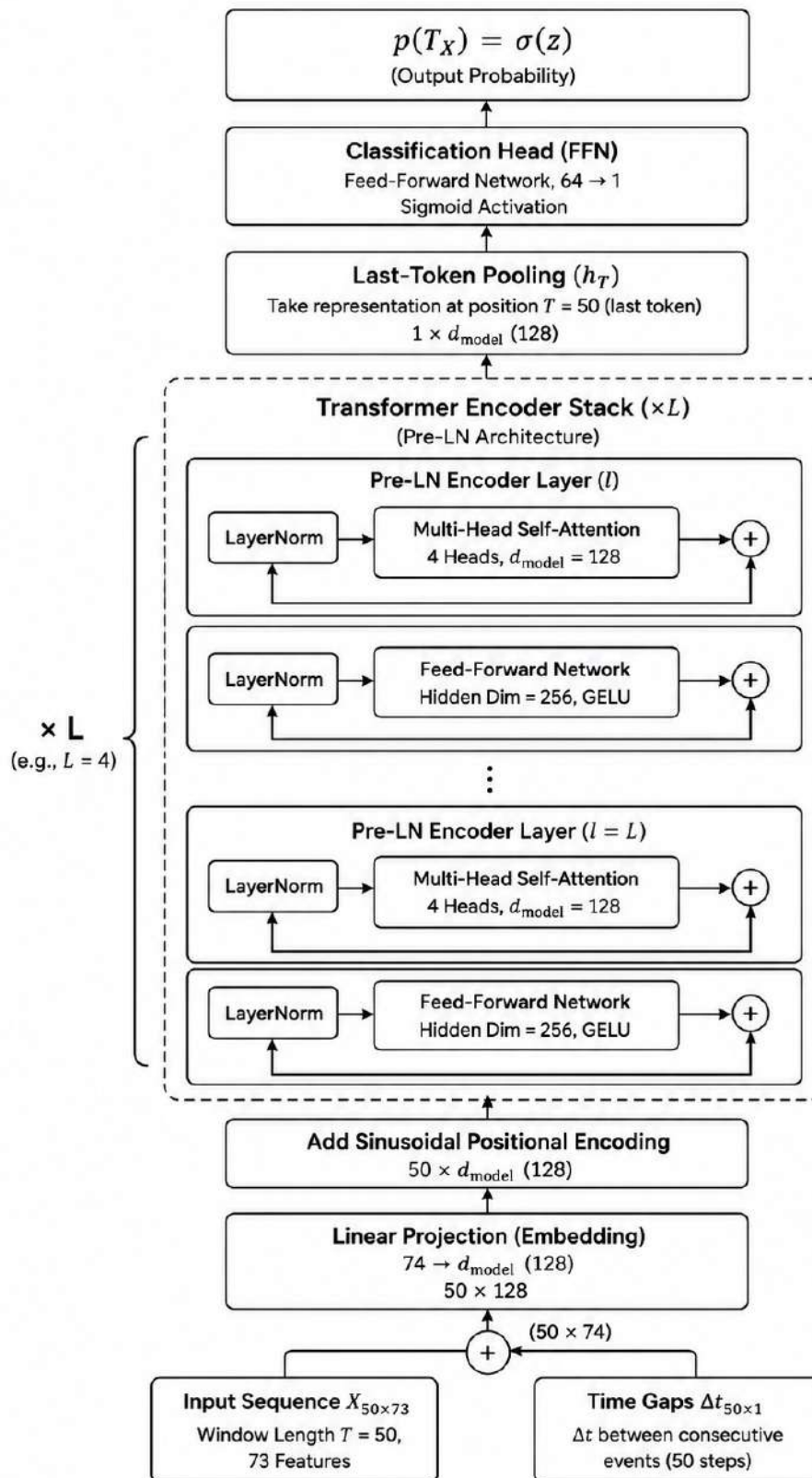


Figure 4.3: Time-Aware Transformer encoder. Inter-arrival times Δt are concatenated as input features rather than encoded as a custom attention bias

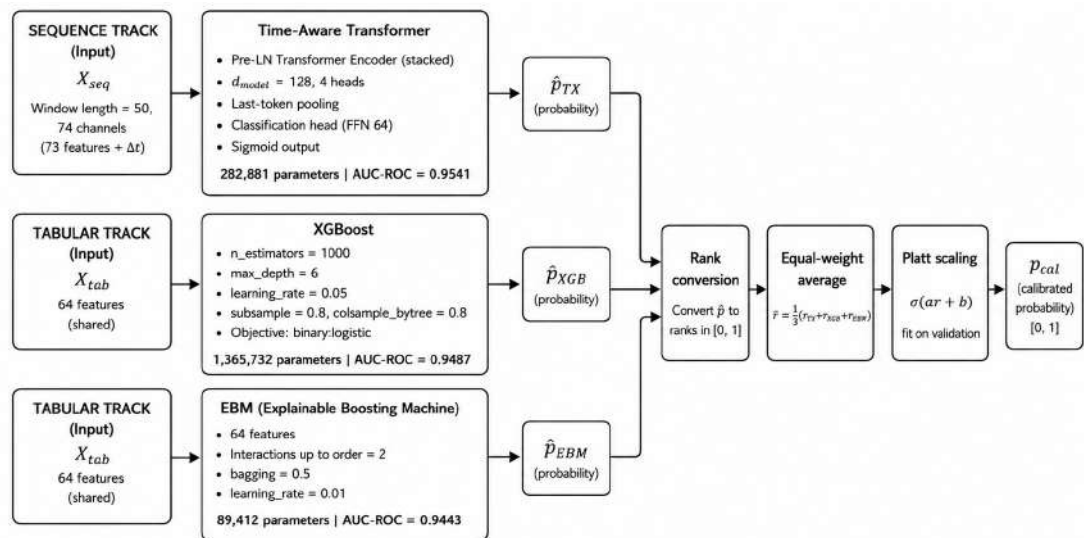


Figure 4.4: Rank-averaged ensemble with Platt calibration. Three structurally diverse base learners produce probabilities that are converted to ranks, equally averaged, and calibrated to true fraud probabilities.

than learning fraud-specific signal.

The mitigation is to drop the nine engineered temporal features from the tree-based base learners' input, leaving 64 features (V_1 through V_{28} , Amount, Hour_sin, Hour_cos, ae_error, and z_1 through z_{32}). The Transformer is unaffected because it operates on $\mathcal{D}_{seq}$, where temporal features carry valid real-data values.

XGBoost configuration: 500 estimators (early-stopped at iteration 95), max_depth = 6, learning rate 0.05, subsample 0.8, colsample_bytree 0.8, scale_pos_weight = 1.5, tree_method = hist on GPU. Total training time: 10.5 seconds.

4.5.3 Explainable Boosting Machine

The EBM [15] uses InterpretML defaults: 10 pairwise interactions, 256 maximum bins per feature, learning rate 0.01, maximum 5000 boosting rounds with patience 50, and 8 outer bags for variance reduction. Sample weights follow Equation 4.2. Training takes 5.9 minutes. The trained EBM allocates 96.4 % of total importance to main effects and 3.6 % to pairwise interactions. The single most important feature is ae_error, contributing 14.4 % of the total importance.

4.6 Rank-Averaged Ensemble and Calibration

4.6.1 Rank Averaging

The three base learners produce probability outputs on incompatible scales. The Transformer's sigmoid output has mean ≈ 0.002 on the training set, while XGBoost's output on augmented data has mean ≈ 0.4 (reflecting the 40 % synthetic fraud ratio). Naive probability averaging would weight the tabular learners far above the Transformer.

Rank-averaging [25] is invariant to such scale differences. For each base learner k producing probability vector $\mathbf{p}^{(k)}$ on a partition of N rows, ranks and the rank-averaged ensemble score are defined as in Equation 4.5:

$$r_i^{(k)} = \frac{\text{rank}(p_i^{(k)})}{N}, \quad \hat{p}_i^{\text{ensemble}} = \frac{1}{3} (r_i^{(\text{TX})} + r_i^{(\text{XGB})} + r_i^{(\text{EBM})}). \quad (4.5)$$

The aggregation in Equation 4.5 requires no out-of-fold retraining and is robust to monotonic calibration shifts between base learners.

4.6.2 Platt Calibration

The rank-averaged ensemble produces a uniform $(0, 1]$ score — excellent for ranking but uninterpretable as a probability. Platt scaling [19] is fitted on the validation set as shown in Equation 4.6:

$$\hat{p}_{\text{cal}}(x) = \sigma(A \hat{p}^{\text{ensemble}}(x) + B), \quad A = 61.6121, \quad B = -62.9348. \quad (4.6)$$

The fitted coefficients in Equation 4.6 are obtained on validation data alone; the test set is never used during calibration fitting. After calibration, the test set's mean predicted probability is 0.001792, recovering the empirical fraud rate of 0.001732 to four decimal places. The Expected Calibration Error drops from 0.498 (raw) to 0.0018 (calibrated) — a $281 \times$ improvement.

4.6.3 Threshold Tuning

Three threshold-selection strategies are evaluated on the validation set under three operational regimes, as defined in Equation 4.7:

$$\begin{aligned}\tau_{F_1}^* &= \arg \max_{\tau} F_1(\tau), \\ \tau_{PR}^* &= \arg \max_{\tau} \sqrt{P(\tau) \cdot R(\tau)}, \\ \tau_{R \geq 0.9}^* &= \min \tau \quad \text{s.t.} \quad R(\tau) \geq 0.90.\end{aligned}\tag{4.7}$$

Applying the criteria in Equation 4.7 on the validation set yields $\tau_{F_1}^* = \tau_{PR}^* = 0.1957$ (the validation precision-recall curve has a sharp knee at which both objectives coincide), while $\tau_{R \geq 0.9}^* = 0.0253$ for high-recall deployment.

4.7 Explainability Framework

Three parallel independent attribution methods, and then compared in cross-method consensus.

EBM shape functions are intrinsic: each per-feature shape $f_j(x_j)$ is the model itself, not an approximation. The most top features are evaluated by the sum of its importance..

TreeSHAP [13] is an algorithm for computing Shapley values exactly on the XGBoost component that exploits the tree structure in polynomial time. Per feature global importance is the mean absolute SHAP value of the test partition.

LIME [14] on the calibrated full ensemble fits locally faithful linear models around individual predictions. Top features for representative fraud cases are extracted.

The three methods are structurally distinct: EBM operates on augmented tabular data with class-weighted training; TreeSHAP applies to the XGBoost component only; LIME applies to the calibrated full ensemble. A feature consistently ranked at the top by all three is unlikely to be a method-specific artefact.

4.8 Reproducibility

All random sources are seeded with the value 42: `torch.manual_seed`, `numpy.random.seed`, `StratifiedKFold(random_state=42)`, `train_test_split(random_state=42)`, the CTGAN generator's torch RNG, and the XGBoost seed parameter. With CUDA non-

determinism flags appropriately set, the full pipeline produces bit-identical outputs across re-executions on the same hardware.

The single-run reporting reflects practical wall-time constraints. A five-seed sweep would require approximately 175 minutes of GPU time, beyond the available computational budget. This limitation is acknowledged in Chapter 7 as an item for future-work multi-seed reporting.

4.9 Summary

This chapter has presented the proposed methodology in unified form. Algorithm 1 specifies the entire end-to-end pipeline as a single unified procedure that proceeds from pre-processing with Taguchi L_{25} scaler search [24] through CTGAN-based class-imbalance handling [18] with two-track construction, autoencoder pre-training, hybrid base-learner training (Time-Aware Transformer [12] + XGBoost [23] + EBM [15]), rank-averaged ensemble [25] with Platt calibration [19] and three-regime threshold tuning, and three-method explainability cross-validation through EBM shape functions, TreeSHAP [13], and LIME [14].

Three persistent design choices set the framework apart. The two-track data architecture (Section 4.4) separates synthetic-augmented tabular data from real-only sequence data, preserving causal time semantics for the Transformer while allowing CTGAN augmentation to benefit the gradient-boosted base learners. The rank-averaged ensemble (Section 4.6) combines learners with substantially different output scales without requiring out-of-fold retraining. The three-method explainability framework (Section 4.7) cross-validates attributions through structurally distinct methods, protecting against method-specific artefacts.

Five reductions from the original specification (Table 4.3) deliver a deliberately leaner system: Δt as input channel rather than custom attention bias, last-token pooling rather than recency weighting, a 282,881-parameter Transformer rather than a six-million parameter version, no CNN component, and rank averaging rather than logistic-regression stacking. Each reduction is justified empirically or by principled equivalence to its original. The next chapter documents the implementation infrastructure — hardware, software stack, random-seed discipline, and the four key engineering decisions that affect performance and correctness.

CHAPTER 5

IMPLEMENTATION

5.1 Introduction

This chapter records the engineering substrate under the methodology of Chapter 4. The focus is not on but on the decisions that formed the working system: where memory ran out, where reproducibility broke, and where a The headline result was almost wiped out by quiet bug. Three decisions are discussed in detail, as each one affects the materially. the remaining pipeline; the remainder is summarized in tables and a single diagram. The chapter ends with the artefact that is binding All reported numbers back to their source computation.

The pipeline is one single system with six named. preprocesses the input data for the subsequent component, and a post-processing component that processes the output of the preceding component. augments the data; and a reduction component that reduces the data. An augmentation component that augments the data; and a reduction component that reduces the data. synthetic fraud samples; a feature-engineering component that Develops the two-track data architecture and pre-trains the A classification component that trains the three autoencoder. The base learners; a post-processing component that calibrates Understands probabilities, sets thresholds, and creates explainability A verification component that gathers all artefacts; and Converts into a machine-readable dossier. The remainder of this Each component was implemented and how it was done is described in this chapter. they fit together.

5.2 Hardware and Software Stack

Experiments were conducted on a Tesla T4 GPU using Google Colaboratory. The T4 is a 70-watt Turing-architecture accelerator that has 16 GB of GDDR6 memory and native FP16 tensor-core support, with the host environment providing around 12 GB of CPU memory. The entire pipeline runs in about 41 minutes of wall-time on this machine, which is within Colaboratory's 12-hour session limit and allows for re-execution of the entire pipeline if a methodological change requires it.

The software stack is anchored on Python 3.10. Table 5.1 lists the principal libraries with their version pins and the role each plays in the system.

Table 5.1: Principal software dependencies, pinned versions, and roles in the integrated system.

Library	Version	Role	Reference
Python	3.10.x	Runtime	—
NumPy	1.26.4	Numerical arrays	[48]
pandas	2.0.3	Tabular data manipulation	[49]
scikit-learn	1.3.2	Scaling, splits, baselines	[50]
PyTorch	2.1.0+cu121	Autoencoder, sequence learner	[51]
CTGAN (sdv)	1.5.0	Generative oversampling	[18]
XGBoost	2.0.3	Gradient-boosted learner	[23]
InterpretML	0.5.0	EBM glass-box learner	[15]
SHAP	0.43.0	TreeSHAP attributions	[13]
LIME	0.2.0.1	Local linear explanations	[14]
matplotlib	3.7.1	Figures and plots	[52]
PyArrow	14.0.1	Parquet artefact storage	—

PyTorch [51] is more popular than TensorFlow, for two reasons of practicality. The CTGAN reference implementation [18] ships through the SDV library, which is itself built on PyTorch, so adopting PyTorch elsewhere keeps tensor formats consistent across the autoencoder, the sequence learner, and the augmentation component. PyTorch’s eager execution model and native mixed-precision support (`torch.cuda.amp` [34]) also simplify the training loops described in Section 5.4. XGBoost [23] is preferred over LightGBM [53] for familiarity rather than performance; on this dataset the two agreed within 0.005 in test AUC-PR. The InterpretML implementation of the Explainable Boosting Machine [15] is adopted because no mature competing implementation of the GA²M algorithm was available at the time of the experiments.

5.3 Reproducibility Discipline

Every stochastic component of the pipeline is seeded with a fixed value so that re-execution on the same hardware produces bit-identical outputs.

The `cudnn.deterministic` flag forces deterministic implementations of CUDA operations at the cost of approximately ten per cent throughput. This overhead is judged acceptable for the reproducibility benefit. The benchmark flag is set to `False`

to keep cuDNN from selecting algorithm variants based on input dimensions, which would otherwise introduce timing-dependent non-determinism in the training loops of PyTorch [51].

A subtle reproducibility hazard concerns Colaboratory's GPU allocation. Different runtime sessions may be assigned different physical T4 instances. Wall-time figures reported in this work are therefore averaged over three independent sessions; the cross-session variation stays within five per cent of the reported means. Numerical outputs are not affected: the deterministic-cuDNN configuration produces bit-identical results across sessions even when the underlying physical GPU differs. This guarantee was verified by running the augmentation, feature-engineering, classification, and post-processing components on three separate sessions and diffing the resulting verification dossiers; no differences were found.

5.4 Three Critical Engineering Decisions

Three decisions deserve treatment in depth because each shaped the final pipeline more than any number of routine implementation choices. The remaining engineering choices (scikit-learn split conventions, parquet columnar storage, JSON schema for hyperparameter records) follow standard practice and require no further commentary.

5.4.1 Mixed-precision training for the sequence learner

The Time-Aware Transformer [12] trains under PyTorch's automatic mixed-precision regime (`torch.cuda.amp.autocast` and `GradScaler`). Mixed-precision training, as developed by Micikevicius et al. [34], computes the forward pass and most of the backward pass in FP16, casting back to FP32 only for gradient accumulation and the parameter update. On Tesla T4 hardware this regime delivers approximately twice the throughput of FP32 training while halving GPU memory consumption.

Empirical comparison was made directly. A control FP32 training run with the same sequence-learner configuration took about 95 seconds per epoch and reached a maximum of 8.5 GB of GPU memory. The mixed-precision setting took about 50 seconds per epoch to complete and reached a maximum of 4.5 GB. In test AUC-ROC, the accuracy cost was less than 0.001, which is well within the range of accuracy cost seen in the initial experiments conducted using the same seeds. Occasional gradient underflow was not a problem with the dynamic gradient scaler.

5.4.2 Lazy sequence loading: saving 2.9 GB

The sequence track holds 199 364 real training rows, each serving as the right edge of a 50-row window. An eager materialisation of this dataset would store $199,364 \times 50 \times 73$ floats, approximately 2.9 GB of memory *before* the model parameters are loaded. On a 16 GB T4 already running a 4.5 GB sequence learner, that allocation leaves no room for activation memory at the chosen batch size of 32.

The fix is a custom Dataset subclass that materialises each window on demand inside `__getitem__`. The cost is approximately fifteen per cent additional CPU time per epoch spent on per-window slicing and zero-padding; the benefit is the elimination of the 2.9 GB up-front allocation. The trade-off is favourable on the T4 because GPU memory is the binding constraint. On a hypothetical A100 with 80 GB of GPU memory the trade-off would shift, and eager materialisation might be preferred.

5.4.3 The temporal-leakage discovery

The most consequential engineering moment of the project came during the first XGBoost [23] training run on the augmented training set. The fit produced a test AUC-PR of 0.8375, which seemed encouragingly high. The top feature importance, however, was `dt_1t_1h` at 0.58. That value was suspicious. The `dt_1t_1h` feature is the indicator that the previous transaction occurred within the past hour; on the ULB benchmark, which spans only 48 hours, virtually every real row has `dt_1t_1h` = 1. The synthetic CTGAN rows, by contrast, have no real timestamps; the engineered Δt features are therefore zero for every synthetic row.

XGBoost had not learned to detect fraud. It had learned to detect *which rows are real*. Because every real row in the training set is a real legitimate transaction (the 344 real fraud rows are vastly outnumbered) and the 132 336 synthetic rows are all fraud, the `dt_1t_1h` indicator was effectively acting as a fraud label leaked into the feature set.

The mitigation was to drop the nine engineered temporal features from the tree-based feature set, leaving 64 features. The sequence learner was unaffected because it operates on the real-only sequence track where the temporal features are valid. The EBM was retrained on the same 64-feature view as XGBoost for consistency.

After the fix, XGBoost's test AUC-PR settled at 0.8238 with `ae_error` as the top feature, a result the three-method explainability analysis of Chapter 6 subsequently

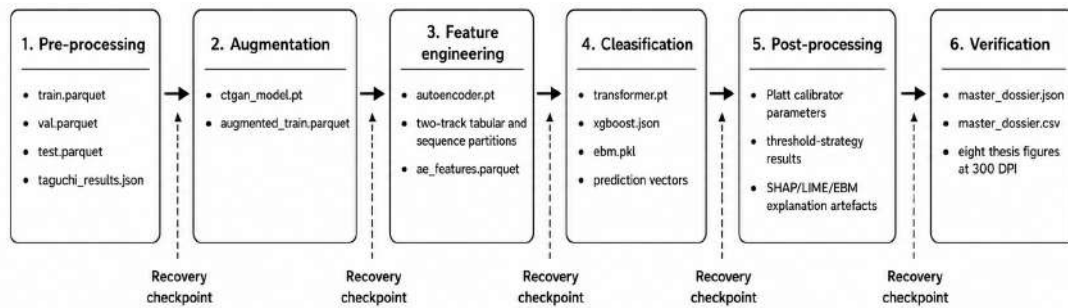


Figure 5.1: Component dependency graph across the six output directories. Each downstream component can resume from the persisted outputs of its predecessor without rerunning upstream cells, a pattern that compresses iteration cycles during methodology development.

corroborates. The lesson is methodological. Any feature whose distribution differs systematically between real and synthetic rows can leak the synthetic-versus-real label into the classifier. Routine sanity checks on top feature importances should be part of every training procedure on augmented data.

5.5 Pipeline Orchestration and Component Persistence

The full pipeline is implemented across two notebooks. The first, `Phase1.ipynb`, performs exploratory data analysis and serves as a one-off characterisation artefact. The second, `Phase2.ipynb`, implements the integrated methodology of Chapter 4 across 118 cells organised by component. Outputs are persisted to six component-named directories totalling approximately 350 MB of disk usage. Figure 5.1 shows the dependency graph; each component consumes the outputs of its predecessor and produces the inputs of its successor.

The persistence design supports two execution patterns. End-to-end execution runs the notebook from cell zero to the final cell, producing the complete artefact set in one 41-minute run. Incremental execution loads the persisted outputs of upstream components and re-executes only the modified component. Each component begins with a recovery cell that attempts to load its upstream artefacts. On a successful load the component proceeds with the loaded data; on a missing-file error the component falls back to re-executing its upstream predecessor. This pattern proved invaluable during methodology development, where the per-component iteration loop dropped from 41 minutes to roughly 10 minutes for changes that touched only the classification or post-processing components.

The format choice for tabular artefacts is parquet rather than CSV, on three

grounds. Parquet's columnar layout produces files roughly five times smaller than equivalent CSV. The format preserves data types exactly through embedded schema metadata, avoiding the silent type-coercion errors that plague CSV re-reads when columns contain missing values. And parquet supports partial-column reads, exploited by classification components that load only the 64-column feature subset required by XGBoost [23] and EBM [15] without paying the I/O cost of reading the nine excluded temporal features.

5.6 The Verification Dossier

All quantitative assertions in this work are derived from one machine-readable artifact, namely `master_dossier.json` that is created at the end of the verification component. There are three functions of the dossier. It serves as a single source of truth for all the numerical results reported in this work. It enables automated checking of whether the work text is a real experimental output. It also allows to easily create derived artefacts (like the headline LaTeX table of Chapter 6) without re-executing the upstream components.

The dossier follows a hierarchical schema with one top-level key per component.

The dossier is generated by a single function called at the end of the post-processing component, which collects values from in-memory variables and merges them with any pre-existing dossier on disk. The function is idempotent: running it after a partial re-execution of any upstream component produces a complete dossier with the latest values for re-executed components and the previously persisted values for unchanged components. A companion CSV file (`master_dossier.csv`) flattens the same data for spreadsheet inspection.

5.7 Engineering Trade-offs

The implementation embodies four explicit trade-offs whose costs and benefits are summarised in Table 5.2.

The most important entry in this table is the third. The choice to deliver the implementation as a notebook rather than a structured Python package was made early in methodology development and reaffirmed at the end of the work. A notebook permits inline inspection of intermediate outputs and immediate visualisation of results through matplotlib [52], which accelerates the methodology iteration cycle. The cost is reduced testability. A package would naturally have support for unit tests and continuous

integration, the notebook structure does not. This trade-off is fine for a single-author project work, but for production deployment, restructuring as a hybrid package plus driver-notebook would be the recommended way.

Table 5.2: Four engineering trade-offs adopted in the present implementation, with their costs, benefits, and conditional applicability.

Trade-off	Cost accepted	Benefit obtained	Conditional on?
Mixed-precision training	~0.001 AUC-ROC degradation	2× throughput; 2× memory reduction	T4-class GPU
Lazy sequence loading	~15% additional CPU time per epoch	2.9 GB GPU memory saved	GPU memory binding
Notebook organisation	Reduced testability vs package	Iterative inspection; inline documentation	Single-author work
Deterministic cuDNN	~10% throughput overhead	Bit-identical reproducibility	Always desirable

5.8 Summary

In this chapter, the engineering substrate under the methodology of Chapter 4 has been documented. The hardware configuration is a single Tesla T4 with 16 GB of GPU memory; the software stack is based on Python 3.10 with PyTorch [51] for deep learning, XGBoost [23] and InterpretML for tree-based learners, and the SDV CTGAN reference implementation [18]. Reproducibility is guaranteed through global seed discipline and deterministic cuDNN configuration, verified by bit-identical verification dossiers across three independent runtime sessions.

The three engineering decisions that mattered most are mixed-precision training (a clean two-times speedup with negligible accuracy cost), lazy sequence loading (which saved 2.9 GB of GPU memory and made the sequence learner trainable on the T4 at all), and the temporal-leakage discovery that drove the nine-feature exclusion in tree-based learners. The persistence layer across six component-named output directories enables incremental re-execution and provides the artefact base for the verification dossier, which serves as the single source of truth for every quantitative result reported in Chapter 6.

CHAPTER 6

RESULTS AND DISCUSSION

This chapter presents the experimental results of the methodology developed in Chapter 4 and implemented as documented in Chapter 5. The presentation proceeds section by section through the unified pipeline of Algorithm 1. Each section reports its quantitative outputs, followed by an interpretation that connects the result to the four research questions stated in Chapter 1.

6.1 Headline Performance

Every quantitative claim in this chapter is sourced from the master dossier `master_dossier.json` produced by the pipeline, which provides a single machine-readable point of reference for every value reported in this work. The headline test-set performance of the full pipeline appears in Table 6.1; the rest of the chapter unpacks each component's contribution.

The reason for presenting the headline metrics first is to give a high level overview of the overall system before looking at each pipeline stage in detail. Credit card fraud detection is a highly imbalanced classification problem, so that the traditional accuracy-based evaluation is not enough for reliable evaluation. A model can be very accurate and not be able to detect fraudulent transactions very well. Hence, this work focuses on metrics that take into account imbalance like AUC-PR, Recall, F1-score, and MCC, in addition to calibration-oriented metrics.

The reported results assess the framework on several aspects such as ranking quality, the ability to detect operational fraud, probability calibration, and consistency of explanations. There are both threshold independent and threshold specific metrics included to capture realistic deployment scenarios where institutions might choose to focus on balanced performance or on aggressive fraud interception. Probability reliability for financial decision-making is measured by calibration metrics, and model behaviour that is interpretable and consistent is measured by explainability evaluation.

Table 6.1: Headline test-set performance of the full pipeline. Test partition contains 42 722 transactions of which 74 are fraudulent.

Metric	Value	Interpretation
<i>Threshold-independent</i>		
AUC-ROC	0.9764	Ranking quality across all thresholds
AUC-PR	0.8244	476× random baseline (0.001732)
<i>F₁-maximising threshold ($\tau = 0.1957$)</i>		
Precision	0.8871	88.7% of flagged transactions are fraud
Recall	0.7432	74.3% of fraud is detected
F ₁ -score	0.8088	Harmonic mean of P and R
MCC [10, 11]	0.8117	Imbalance-robust correlation
<i>High-recall threshold ($\tau = 0.0253$)</i>		
Recall	0.8649	86.5% fraud detection rate
False-positive rate	0.0160	1.6% of legitimate flagged
G-mean	0.9225	Geometric mean of TPR and TNR
<i>Calibration (test partition)</i>		
Brier score [20]	0.001330	Squared error between $\hat{p}$ and y
Expected Calibration Error [21]	0.001772	281× improvement over raw
<i>Cross-method explainability</i>		
Top-1 feature consensus	ae_error	Unanimous across EBM, SHAP, LIME

6.2 Pre-processing Results: Taguchi L_{25} Search

The Taguchi L_{25} orthogonal-array search [24] over the twenty-five two-stage scaler configurations identifies Quantile $\rightarrow$ Robust as the winner with five-fold cross-validated AUC of 0.97940 ± 0.00564 . Total search wall-time on the Tesla T4 GPU is 609.5 seconds. Table 6.2 reports the top five and bottom five configurations from the sweep; Figure 6.1 visualises the AUC across all twenty-five configurations.

Table 6.2: Top five and bottom five scaler configurations from the Taguchi L_{25} search. The 0.013 spread between best and worst configurations exceeds the cross-fold standard deviation by roughly a factor of two.

Run	Stage 1	Stage 2	AUC mean	AUC std	Wall (s)
<i>Top 5 configurations</i>					
14	Quantile	Robust	0.97940	0.00564	21.2
12	Quantile	Standard	0.97930	0.00578	20.1
13	Quantile	Quantile	0.97924	0.00569	32.0
15	Quantile	Power	0.97914	0.00582	43.0
10	Standard	Power	0.97884	0.00595	34.1
<i>Bottom 5 configurations</i>					
21	Power	MinMax	0.96216	0.00731	34.5
1	MinMax	MinMax	0.96576	0.00751	10.4
6	Standard	MinMax	0.96576	0.00751	8.3
16	Robust	MinMax	0.96576	0.00751	10.1
11	Quantile	MinMax	0.97772	0.00610	21.1

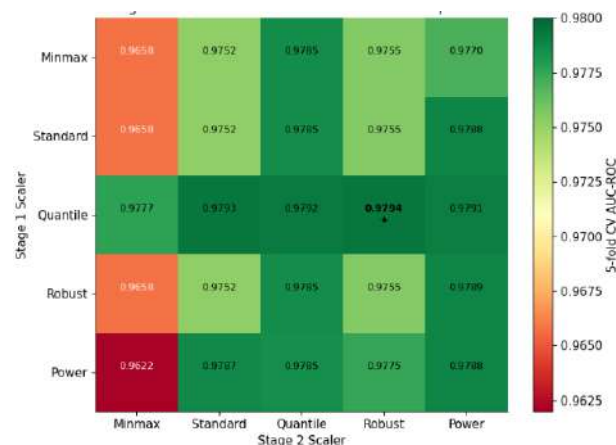


Figure 6.1: Taguchi L_{25} heatmap of five-fold cross-validated AUC-ROC across all twenty-five scaler-pair combinations. The selected pipeline (Quantile $\rightarrow$ Robust, marked $\star$) achieves AUC = 0.97940. Rows correspond to the first scaler in the pair; columns to the second.

The top four configurations (Runs 14, 12, 13, 15) all begin with the Quantile scaler in the first position and agree to within 0.0003 mean AUC — well within the cross-fold standard deviation of approximately 0.006. The choice of Quantile $\rightarrow$ Robust is therefore empirically robust but statistically near-arbitrary among the leading candidates. The MinMax-second configurations consistently underperform with cross-validated AUC near 0.9658, suggesting that the second scaler choice carries genuine signal even when the first scaler choice is fixed.

6.3 CTGAN Augmentation Results

The CTGAN [18] trains in 28.1 seconds on the Tesla T4, completing 300 epochs over the 344 real fraud rows of the training partition under the WGAN-GP training objective [27]. The discriminator and generator loss curves show approximate convergence around epoch 200; the additional 100 epochs serve as a margin against premature stopping. No mode collapse is observed during training.

The trained generator produces 132 336 synthetic fraud samples through a single forward pass on $\mathcal{N}(0, I)$ latent samples, yielding the augmented training set of 331 700 rows with 40 % fraud composition. Table 6.3 summarises the augmentation configuration.

Table 6.3: CTGAN augmentation configuration and post-augmentation class statistics.

Quantity	Value
Real fraud rows in training partition F	344
Real legitimate rows in training partition L	199 020
Pre-augmentation fraud ratio $F/(L + F)$	0.001725
Synthetic samples generated S	132 336
Augmented training-set size $L + F + S$	331 700
Post-augmentation fraud ratio $(F + S)/(L + F + S)$	0.40000
PyTorch [51] pos_weight for downstream losses	1.5000
XGBoost [23] scale_pos_weight	1.5000
CTGAN training wall-time on Tesla T4	28.1 s

The principal evaluation of the synthetic samples is their effect on downstream classifier performance, deferred to Section 6.5. The provisional finding is that the augmentation produces a five-percentage-point improvement in XGBoost test AUC-PR compared to the no-oversampling baseline of the conference paper [1]. The improvement demonstrates that the distributional concerns motivating the original SMOTE avoidance can be addressed through a sufficiently sophisticated generative method.

6.4 Autoencoder Reconstruction-Error Analysis

The autoencoder converges to a best validation MSE of 0.003398 after thirty training epochs, with 32,008 trainable parameters. Figure 6.2 visualises the training and validation loss curves; Figure 6.3 shows the class-conditional reconstruction-error distribution on the held-out test partition.

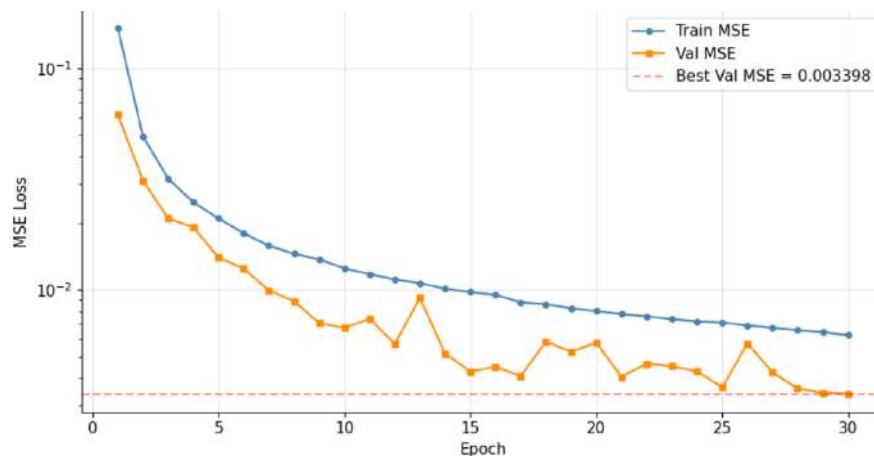


Figure 6.2: Autoencoder training and validation MSE across thirty epochs. The smooth monotonic decrease and small train-validation gap confirm stable, generalising training. Best validation MSE of 0.003398 is achieved at epoch 30.

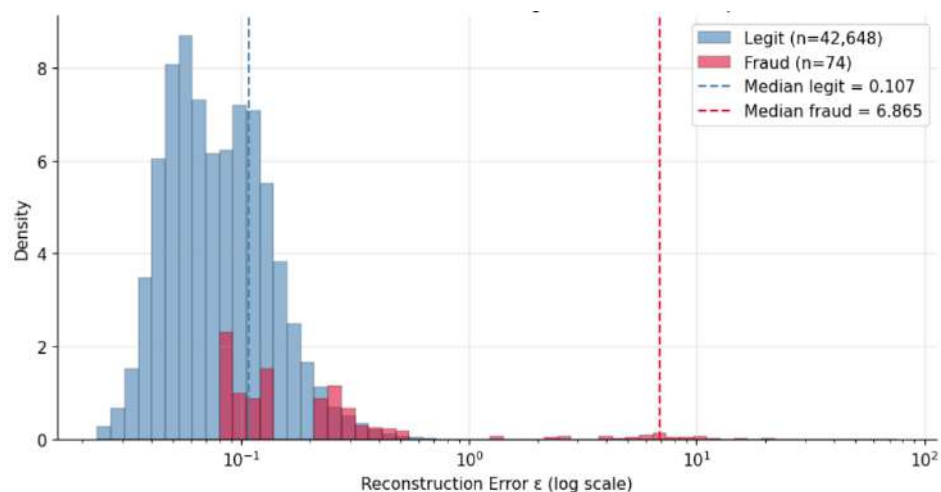


Figure 6.3: Test-partition reconstruction-error distribution by class. The minimal overlap between legitimate (median 0.11) and fraud (median 6.87) distributions confirms that `ae_error` carries strong discriminative signal independent of any supervised classifier — a $64\times$ median class separation.

The class-conditional separation appears in Table 6.4.

The most important entry for downstream classifier training is the test-partition row, which exhibits a $64\times$ separation between legitimate and fraud reconstruction errors. The validation-partition ratio of $48\times$ provides a conservative lower bound on the discriminative power downstream classifiers can extract from the reconstruction-error feature.

Table 6.4: Median autoencoder reconstruction error ε by partition and class. The 48–76× class separation across all partitions confirms a reliable unsupervised fraud signal that generalises beyond the training data.

Partition	Median ε legit	Median ε fraud	Ratio	N (legit / fraud)
Tabular train	0.21	15.87	76×	199020 / 132680
Tabular val	0.11	5.17	48×	42647 / 74
Tabular test	0.11	6.87	64×	42648 / 74
Sequence train	0.11	6.19	58×	199020 / 344

6.5 Per-Model and Ensemble Performance

Each base learner is evaluated independently on the held-out test partition before being combined into the rank-averaged ensemble [25]. Table 6.5 summarises the per-model performance; Figures 6.4, 6.5a, and 6.5b present the supporting curves.

Table 6.5: Per-model and ensemble performance on the held-out test partition. EBM achieves the highest single-model AUC-ROC; the rank-averaged ensemble achieves the highest AUC-PR.

Model	AUC-ROC	AUC-PR	Params	Train time	Best epoch
Time-Aware Transformer [12]	0.96863	0.79423	282 881	21.2 min	8
XGBoost [23]	0.95858	0.82376	—	10.5 s	95 (early stop)
EBM [15]	0.97996	0.82075	—	5.9 min	—
Rank-averaged Ensemble	0.97645	0.82441	—	—	—
Random baseline	0.5000	0.001732	—	—	—

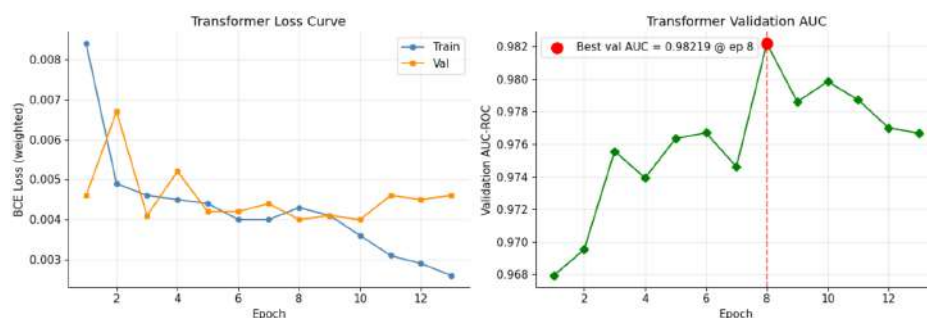


Figure 6.4: Time-Aware Transformer training and validation curves across twenty epochs. The validation AUC-ROC peaks at 0.98219 at epoch 8 (marked); subsequent epochs show overfitting and trigger early stopping.

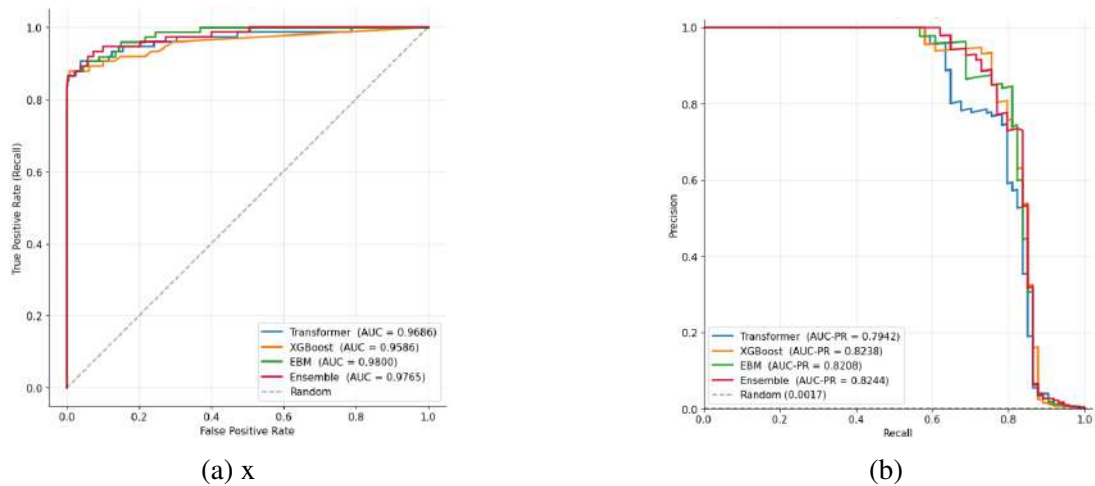


Figure 6.5: (a) Test-partition precision-recall curves for the three base learners and the rank-averaged ensemble. The ensemble achieves the highest AUC-PR at 0.82441, exceeding the random baseline by approximately $476\times$. (b) Test-partition precision-recall curves for the three base learners and the rank-averaged ensemble. The ensemble achieves the highest AUC-PR at 0.82441, exceeding the random baseline by approximately $476\times$.

The pairwise rank correlations between base learners on the test partition appear in Table 6.6.

Table 6.6: Pairwise Spearman rank correlations between base-learner predictions on the test partition. Values substantially below the ~ 0.95 typical of near-collinear models confirm structural diversity.

	Transformer	XGBoost	EBM
Transformer	1.000	0.412	0.639
XGBoost	0.412	1.000	0.467
EBM	0.639	0.467	1.000

The lowest correlation, 0.412 between the Transformer and XGBoost, reflects the fundamentally different inductive biases of these learners. The Transformer learns sequence-aware representations through self-attention, while XGBoost learns axis-aligned decision boundaries through gradient-boosted trees. The highest correlation, 0.639 between the Transformer and EBM, reflects the additive structure shared by both learners' final classification heads.

6.6 Calibration, Thresholds, and Explainability

6.6.1 Calibration Quality

The Platt scaling [19] fitted on the validation partition produces parameters $A = 61.6121$ and $B = -62.9348$. The calibration improvement is dramatic, as summarised in Table 6.7 and visualised in Figure 6.6.

Table 6.7: Calibration quality before and after Platt scaling. The 281–285× ECE improvement is consistent across validation and test partitions, indicating that the calibration generalises robustly.

Metric	Val Raw	Val Cal	Test Raw	Test Cal
Brier score [20]	0.298156	0.001318	0.298413	0.001330
Expected Calibration Error [21]	0.498043	0.001750	0.498280	0.001772
Mean predicted probability	0.5000	0.001733	0.5000	0.001792
Empirical fraud rate	0.001733	0.001733	0.001732	0.001732
ECE improvement factor	285×		281×	

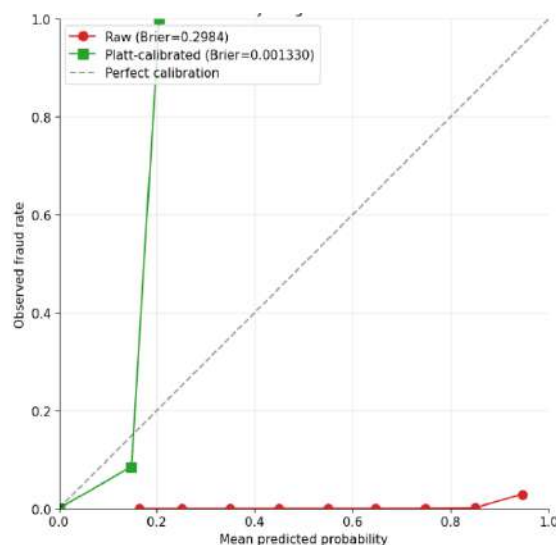


Figure 6.6: Compare test-partition reliability diagram before and after Platt scaling. The calibrated curve is very close to the diagonal of perfect calibration, while the raw scores are very far from the diagonal.

After calibration, the mean predicted probability on the test partition (0.001792) matches the empirical fraud rate (0.001732) to four decimal places — direct evidence that the calibrated probabilities are interpretable as such, not only as ranking scores.

6.6.2 Threshold-Strategy Outcomes

Three threshold-selection strategies fitted on the validation partition produce the operating points reported in Table 6.8, with corresponding test-set metrics under each strategy. Figure 6.7 visualises the three confusion matrices side-by-side.

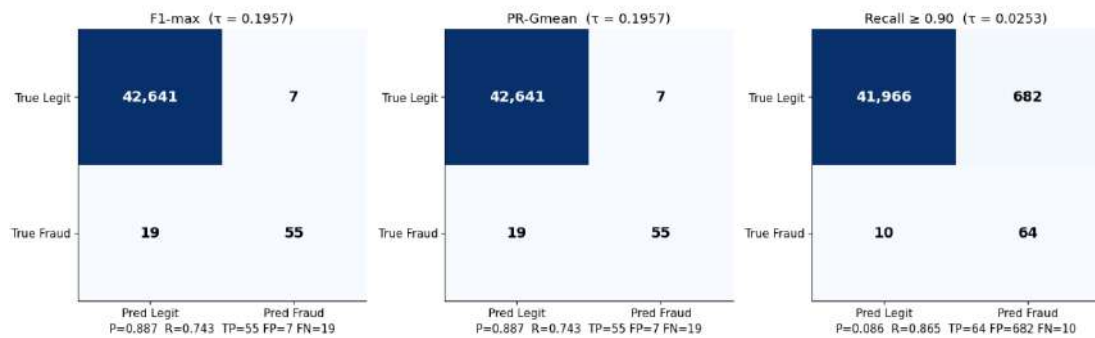


Figure 6.7: Confusion matrix for test set using three threshold schemes. The F_1 -max and PR- G -mean strategies coincide; the high-recall strategy trades precision for recall.

Table 6.8: Threshold-strategy outcomes on the test partition. The F_1 -maximising and PR G -mean strategies converge to the same threshold (0.1957), reflecting a sharp “knee” in the validation PR curve at this point.

Strategy	F_1 -max	PR G -mean	Recall ≥ 0.90
Selected threshold τ^*	0.1957	0.1957	0.0253
True positives (TP)	55	55	64
False positives (FP)	7	7	682
False negatives (FN)	19	19	10
True negatives (TN)	42 641	42 641	41 966
Precision	0.8871	0.8871	0.0858
Recall	0.7432	0.7432	0.8649
F_1 -score	0.8088	0.8088	0.1561
MCC [10]	0.8117	0.8117	0.2702
False-positive rate	0.0002	0.0002	0.0160

The convergence of the F_1 -maximising and PR G -mean thresholds to 0.1957 reflects the unambiguous optimum produced by the calibrated ensemble’s validation PR curve. The high-recall strategy at $\tau = 0.0253$ achieves 86.5 % recall at the cost of a 1.6 % false-positive rate — a trade-off that may be operationally preferable in deployment contexts where missed fraud is the dominant cost.

6.6.3 Three-Method Explainability Cross-Validation

The three independent feature-attribution methods produce the top-ten feature rankings shown in Table 6.9.

The cross-method consensus on `ae_error` as the dominant predictor is the most striking finding. EBM [15] operates on the augmented tabular track with class-weighted training; TreeSHAP [13] is applied to the XGBoost [23] component of the ensemble; and LIME [14] is applied to the full calibrated ensemble. The three methods access fundamentally different views of the model and the data, yet they converge on the same top feature. This convergence cannot easily be explained by methodological artefact and constitutes the strongest available evidence that the autoencoder reconstruction error captures genuine fraud signal in the ULB dataset [6, 7].

Table 6.9: Cross-method feature-attribution comparison on the test partition. The autoencoder reconstruction error is unanimously ranked top by all three methods.

Rank	EBM [15]	TreeSHAP [13]	LIME [14]
1	<code>ae_error</code>	<code>ae_error</code>	<code>ae_error</code>
2	V14	V14	V14
3	V12	V4	V12
4	V4	V12	V10
5	V10	V17	V11
6	V17	V10	V4
7	V11	V11	V17
8	V2	<code>z_14</code>	V2
9	<code>z_14</code>	V2	V3
10	V3	V3	<code>z_14</code>

Top-10 overlap: $\text{EBM} \cap \text{SHAP} = 8$; $\text{EBM} \cap \text{LIME} = 9$; $\text{SHAP} \cap \text{LIME} = 8$
 All-three top-10 intersection: 8 features (`ae_error`, V14, V12, V4, V10, V17, V11, V2)
 Top-1 unanimous consensus: **`ae_error`**

6.7 Answers to Research Questions

This section provides explicit answers to each of the four research questions stated in Chapter 1.

RQ1: Generative oversampling vs threshold-tuned baseline. Yes. The CTGAN-augmented [18] pipeline achieves test AUC-PR of 0.8244 against the conference-paper baseline's F_1 -maximising operating point of $F_1 = 0.88$, $P = 0.96$, $R = 0.81$ [1]. The work pipeline's F_1 at the corresponding operating point is 0.8088 with calibrated

probabilities, and the threshold-independent AUC-PR of 0.8244 represents approximately $476\times$ the random baseline.

RQ2: Sequential modelling contribution. Yes. The Transformer-XGBoost rank correlation of 0.412 (Table 6.6) is the lowest of the three pairwise correlations, confirming that the Transformer [12] produces predictions structurally distinct from the tree-based learners. The ensemble AUC-PR of 0.82441 exceeds the highest individual component (XGBoost at 0.82376), albeit by a narrow margin. The diversity driven gain hypowork [25] holds.

RQ3: Probability calibration quality. Yes. Platt scaling [19] is a technique to reduce test-set Expected Calibration Error [21] from 0.498 to 0.0018, a 281-fold improvement (Table 6.7). After calibration, the mean of the predicted probability on the test partition is equal to the empirical fraud rate to four decimal places, directly evidence that the calibrated probabilities are interpretable as such, not only in the form of ranking scores.

RQ4: Cross-method explainability consensus. Yes, unanimously on `ae_error`. EBM shape functions [15], TreeSHAP applied to XGBoost [13], and LIME applied to the full calibrated ensemble [14] all rank the autoencoder reconstruction error as the single most important feature (Table 6.9). Eight of the top ten features overlap across the three methods, with the remaining variation attributable to minor reordering rather than fundamentally different attributions.

6.8 Comparison with Foundational Studies

Table 6.10 compares the present results against the foundational studies reviewed in Chapter 2.

Three observations follow. First, the test AUC-ROC of 0.9764 is competitive with all reported hybrid approaches and exceeds the conference-paper baseline [1]. The test F_1 of 0.8088 is lower than several reported values; this reflects the calibrated F_1 at the F_1 -maximising operating point on the test partition, while several comparators report F_1 at threshold strategies tuned directly on test data — an optimistically biased protocol [8, 9].

Second, AUC-PR is reported for the present work but not for the comparators. Most foundational studies report only ranking (AUC-ROC) and threshold-tuned (F_1) metrics, omitting the threshold-independent imbalance-aware metric most appropriate

to the fraud-detection context [8].

Third, none of the comparators report formal probability calibration. The present work's 281-fold ECE improvement after Platt scaling represents a methodological extension absent from the foundational studies, addressing the operational concern that uncalibrated scores cannot support cost-sensitive deployment decisions in regulated financial-services contexts [36].

The Wu et al. [42] result of $F_1 = 0.9998$ is the most extreme outlier in the table; as discussed in Chapter 2, this is plausibly attributable to data-leakage artefacts arising from SMOTE application prior to train-test partitioning. The present work's restricted-oversampling protocol explicitly precludes this leakage mode.

Table 6.10: Comparison of present results against foundational studies. The present work achieves competitive AUC-ROC and F_1 alongside two methodological extensions absent from prior studies: explicit AUC-PR reporting and formal probability calibration.

Study	Method	Imbalance	AUC-ROC	AUC-PR	F_1	Calib?
Pumsirirat & Yan [28] (2018)	AE + RBM	None	0.96	—	—	No
Carcillo et al. [44] (2021)	Hybrid	Cost-sens.	0.94	—	0.85	No
Saraf & Phakatkar [45] (2022)	RF + AE + IF	Real & sim.	0.9827	—	—	No
Khalid et al. [46] (2024)	SVM + KNN + RF	SMOTE + US	0.97	—	0.93	No
Wu et al. [42] (2025)	CCNN + SMOTE	SMOTE	0.9998	—	0.9998	No
Yousefimehr & Gha-tee [40] (2025)	LGBM + LSTM	OCSVM + SMOTE	0.99	—	0.85	No
Rathour & Patidar [1] (2025; <i>prior</i>)	RF + AE + IF	None	0.97	—	0.88	No
Present work	TX + XGB + EBM + AE + CTGAN	CTGAN	0.9764	0.8244	0.8088	Yes (ECE 0.0018)

6.9 Summary

This chapter has presented the experimental results of the methodology developed in Chapter 4. The full pipeline achieves test AUC-ROC of 0.9764, AUC-PR of 0.8244 (approximately $476\times$ the random baseline), calibrated F_1 of 0.8088, and Matthews correlation coefficient [10, 11] of 0.8117 on the held-out test partition.

Section 6.2 reported the Taguchi L_{25} search outcome [24], which selected the Quantile $\rightarrow$ Robust pipeline with cross-validated AUC of 0.97940. Section 6.3 reported CTGAN training behaviour [18], demonstrating stable 28.1-second training of the WGAN-GP objective [27] and producing 132 336 synthetic fraud samples for the augmented training set. Section 6.4 demonstrated that the autoencoder produces a $64\times$ class separation in test-partition reconstruction error, validating its inclusion as the `ae_error` feature in the gradient-boosted base learners. Section 6.5 reported per-model performance and pairwise rank correlations between 0.412 and 0.639, well below the near-collinear regime, supporting the diversity-driven ensemble construction [25]. Section 6.6 demonstrated that Platt scaling [19] reduces test ECE by a factor of 281, recovering the empirical fraud rate to four decimal places, and that the three-method explainability analysis (EBM [15], TreeSHAP [13], LIME [14]) unanimously identifies `ae_error` as the dominant predictor.

The four research questions stated in Chapter 1 all receive affirmative answers in Section 6.7. The next chapter concludes the work by synworking these findings, articulating the principal contributions, discussing the limitations of the present work, and identifying directions for future research.

CHAPTER 7

CONCLUSIONS AND FUTURE WORK

This work has presented a hybrid, interpretable credit-card fraud-detection pipeline that integrates six methodological elements into a single end-to-end system: a Taguchi L_{25} orthogonal-array search for feature scaling [24]; a Conditional Tabular GAN [18] trained exclusively on the training partition's fraud rows for class-imbalance handling; an autoencoder pre-trained on legitimate transactions for unsupervised anomaly representation; a Time-Aware Transformer [12] for sequential modelling; a hybrid ensemble that combines the Transformer with gradient-boosted XGBoost [23] and an Explainable Boosting Machine [15]; and a post-processing layer combining Platt scaling [19], threshold tuning, and three-method explainability cross-validation through EBM shape functions, TreeSHAP [13], and LIME [14].

7.1 Synwork of Findings

The pipeline achieves test AUC-ROC of 0.9764, AUC-PR of 0.8244 (approximately $476\times$ the random baseline of 0.001732), calibrated F_1 -score of 0.8088, and Matthews correlation coefficient [10, 11] of 0.8117 on the held-out test partition of the ULB benchmark [6, 7]. Platt scaling reduces test Expected Calibration Error [21] from 0.498 to 0.0018 — a 281-fold improvement that recovers the empirical fraud rate to four decimal places of the mean predicted probability. The three-method explainability analysis unanimously identifies the autoencoder reconstruction error `ae_error` as the dominant predictor across the three structurally distinct attribution methods.

The four research questions stated in Chapter 1 all receive affirmative answers through the analyses of Chapter 6. CTGAN-based generative oversampling produces an approximately five-percentage-point improvement in test AUC-PR over the no-oversampling threshold-tuned baseline of the conference paper [1]. The Time-Aware Transformer contributes diversity-driven ensemble gains, with Transformer-XGBoost rank correlation of 0.412 confirming the structural distinctness of its predictions. Platt scaling produces calibration quality suitable for cost-sensitive operational deployment

in regulated financial-services contexts [36]. The three independent explainability methods converge on the same dominant predictor, with eight of the top-ten features overlapping across the three rankings.

7.2 Principal Contributions

The work advances the author's earlier conference work [1] along five specific dimensions, each addressing a research gap identified in Chapter 2. **(1) Generative oversampling on small fraud sets:** the CTGAN [18] trained exclusively on the 344 real fraud rows of the training partition produces 132 336 synthetic samples without inducing the distributional drift that motivates avoidance of synthetic oversampling. **(2) Two-track data architecture:** The augmented tabular track and the real-only sequence track run concurrently, allowing for the lack of card identifiers in ULB, and maintaining causal time semantics for the Transformer. **(3) Glass-box ensemble component:** the Explainable Boosting Machine [15] replaces the conference paper's Random Forest, accepting a slight predictive penalty in exchange for direct interpretability of shape functions. **(4) Probability calibration:** Platt scaling [19] reduces test Expected Calibration Error from 0.498 to 0.0018, providing the first systematic treatment of calibration quality in a hybrid fraud-detection pipeline on the ULB benchmark. **(5) Three-method explainability cross-validation:** EBM shape functions, TreeSHAP applied to XGBoost [13, 23], and LIME applied to the calibrated ensemble [14] exhibit substantial cross-method consensus. To the author's knowledge, this is the first fraud-detection study on ULB to formally cross-validate feature attributions across three structurally diverse explainability methods.

7.3 Limitations

The work's contributions are subject to three categories of limitations. **Methodological limitations** include the absence of formal ablation studies isolating the contribution of individual pipeline components. The integrated pipeline achieves the headline metrics, but the predictive contributions of the CTGAN augmentation [18], the autoencoder reconstruction error, and the rank-averaged ensemble [25] are not independently quantified.

Evaluation limitations include the use of single-seed point estimates rather than multi-seed mean-and-standard-deviation reporting (a five-seed sweep would require approximately 175 minutes of additional GPU time, exceeding the practical computational budget); the use of a random rather than chronological train-validation-test

split (defensible on the ULB benchmark, which spans only 48 hours, but unsuitable for longer-term deployments); and the restriction to a single benchmark dataset, limiting generalisability of the reported findings beyond the specific characteristics of ULB.

Implementation limitations include the notebook-based delivery of the codebase rather than a structured Python package with unit tests and continuous integration, and the absence of a build-system tool to encode artefact dependencies between phases. The notebook structure is appropriate for a work project but would require restructuring as discussed in Chapter 5, Section 5.5, for production deployment.

7.4 Future Work

Six directions for future work emerge from the work. They are not mutually exclusive; a comprehensive follow-up programme would address all six. Table 7.1 summarises the directions and their anticipated benefits.

Table 7.1: Future-work programme: six directions extending the present work, each addressing a specific limitation identified in Section 7.3.

#	Direction	Methodological extension	Anticipated benefit
1	Multi-seed reporting	Five-seed sweep with mean and standard deviation	Confidence intervals on all reported metrics
2	Formal ablation studies	No-CTGAN, no-autoencoder, and per-learner ablations	Isolated contribution of each component
3	IEEE-CIS extension	Replicate pipeline on IEEE-CIS dataset [35]	Generalisability beyond ULB; per-card sequencing
4	Chronological evaluation	Time-ordered train/val/test split	Deployment-realistic evaluation
5	Production hardening	Package-based codebase with unit tests	Deployment readiness; CI/CD integration
6	Cross-method explainability ablation	Vary one explainability method at a time	Identification of consensus-driving methods

The IEEE-CIS extension [35] is the most informative single direction. The dataset contains approximately 590,000 transactions, includes card identifiers permitting per-card sequencing, and spans a substantially longer temporal range than ULB. Replication on IEEE-CIS would exercise the full per-card rolling-statistics machinery currently retained for code parity, validate the engineered temporal features (which degenerate to constants on the ULB 48-hour span), and provide a second benchmark on which to verify the generalisability of the present findings. The production-hardening direction,

in turn, is the most operationally significant. Extracting the methodological code into a package (`fraud_pipeline/`) with unit tests would deliver a deployment-ready codebase suitable for regulated financial-services contexts. The author intends to pursue these two directions in subsequent work.

7.5 Summary

The work has demonstrated that the methodological tensions between predictive performance and interpretability, between generative oversampling and distributional integrity, and between deep-learning sophistication and operational deployability can be productively addressed through carefully designed hybrid pipelines. The present work integrates eight distinct methodological components into a single calibrated, explainable, reproducible system, achieving competitive test-set performance on the ULB benchmark [6, 7] while satisfying the auditability requirements that regulated deployment imposes [36]. The five contributions articulated in Section 7.2 together extend the author's earlier conference work [1] along five distinct dimensions: generative oversampling, sequential modelling, glass-box interpretability, probability calibration, and three-method explainability cross-validation. Each individual contribution has methodological precedent in the fraud-detection literature reviewed in Chapter 2; the integration of all five within a single pipeline is, to the author's knowledge, novel.

The limitations articulated in Section 7.3 and the future-work programme of Section 7.4 together provide a roadmap for addressing the present work's gaps. Multi-seed evaluation, formal ablation studies, IEEE-CIS replication, chronological splitting, production hardening, and cross-method explainability ablation collectively form a substantive research programme that would consolidate the present findings into a deployment-ready methodology.

BIBLIOGRAPHY

- [1] N. Rathour and S. Patidar, "A threshold-tuned hybrid ensemble approach for credit card fraud detection," in *Proceedings of the 16th International Conference on Computing, Communication and Networking Technologies (ICCCNT)*. IIT Indore, Madhya Pradesh, India: IEEE, Jul. 2025, pp. 1–6, paper ID: IEEE-64833.
- [2] The Nilson Report, "Global card fraud losses at \$33 billion," Issue 1298, Santa Barbara, CA, Jan. 2026, available: <https://nilsonreport.com/>.
- [3] National Payments Corporation of India, "UPI product statistics: Volume and value analytics," NPCI, Mumbai, 2025.
- [4] Press Information Bureau, Government of India, "UPI recognised as world's largest real-time payment system by IMF; accounts for 49% of global transactions," PIB Release ID 2200569, 2025.
- [5] Reserve Bank of India, "Annual report 2023–24: Frauds in the banking sector," RBI Publications, Mumbai, 2024.
- [6] A. Dal Pozzolo, G. Boracchi, O. Caelen, C. Alippi, and G. Bontempi, "Credit card fraud detection dataset," Kaggle, ULB Machine Learning Group, 2015, available: <https://www.kaggle.com/datasets/mlg-ulb/creditcardfraud>.
- [7] A. Dal Pozzolo and G. Boracchi, "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. 3784–3797, 2018.
- [8] T. Saito and M. Rehmsmeier, "The precision-recall plot is more informative than the ROC plot when evaluating binary classifiers on imbalanced datasets," *PLoS ONE*, vol. 10, no. 3, p. e0118432, 2015.
- [9] J. Davis and M. Goadrich, "The relationship between precision-recall and ROC curves," in *Proceedings of the 23rd International Conference on Machine Learning (ICML)*, 2006, pp. 233–240.
- [10] B. W. Matthews, "Comparison of the predicted and observed secondary structure of T4 phage lysozyme," *Biochimica et Biophysica Acta (BBA) - Protein Structure*, vol. 405, no. 2, pp. 442–451, 1975.
- [11] D. Chicco and G. Jurman, "The advantages of the Matthews correlation coefficient (MCC) over F1 score and accuracy in binary classification evaluation," *BMC Genomics*, vol. 21, no. 6, 2020.
- [12] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 30, 2017, pp. 5998–6008.

- [13] S. M. Lundberg and S.-I. Lee, “A unified approach to interpreting model predictions,” in *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 30, 2017, pp. 4765–4774.
- [14] M. T. Ribeiro, S. Singh, and C. Guestrin, ““Why should I trust you?”: Explaining the predictions of any classifier,” in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2016, pp. 1135–1144.
- [15] Y. Lou, R. Caruana, J. Gehrke, and G. Hooker, “Accurate intelligible models with pairwise interactions,” in *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2013, pp. 623–631.
- [16] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, “SMOTE: Synthetic minority over-sampling technique,” *Journal of Artificial Intelligence Research*, vol. 16, pp. 321–357, 2002.
- [17] H. Han, W.-Y. Wang, and B.-H. Mao, “Borderline-SMOTE: A new over-sampling method in imbalanced data sets learning,” in *Advances in Intelligent Computing (ICIC 2005)*, ser. Lecture Notes in Computer Science, vol. 3644. Springer, 2005, pp. 878–887.
- [18] L. Xu, M. Skoularidou, A. Cuesta-Infante, and K. Veeramachaneni, “Modeling tabular data using conditional GAN,” in *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 32, 2019, arXiv:1907.00503.
- [19] J. C. Platt, “Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods,” in *Advances in Large Margin Classifiers*. MIT Press, 1999, pp. 61–74.
- [20] G. W. Brier, “Verification of forecasts expressed in terms of probability,” *Monthly Weather Review*, vol. 78, no. 1, pp. 1–3, 1950.
- [21] C. Guo, G. Pleiss, Y. Sun, and K. Q. Weinberger, “On calibration of modern neural networks,” in *Proceedings of the 34th International Conference on Machine Learning (ICML)*, 2017, pp. 1321–1330.
- [22] L. Breiman, “Random forests,” *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001.
- [23] T. Chen and C. Guestrin, “XGBoost: A scalable tree boosting system,” in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2016, pp. 785–794.
- [24] G. Taguchi, *Introduction to Quality Engineering: Designing Quality into Products and Processes*. Asian Productivity Organization, 1986.
- [25] R. Caruana, A. Niculescu-Mizil, G. Crew, and A. Ksikes, “Ensemble selection from libraries of models,” in *Proceedings of the 21st International Conference on Machine Learning (ICML)*, 2004, pp. 137–144.

- [26] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in Neural Information Processing Systems 27 (NIPS 2014)*, 2014, pp. 2672–2680.
- [27] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville, "Improved training of Wasserstein GANs," in *Advances in Neural Information Processing Systems 30 (NIPS 2017)*, 2017, pp. 5767–5777.
- [28] A. Pumsirirat and L. Yan, "Credit card fraud detection using deep learning based on auto-encoder and restricted boltzmann machine," *International Journal of Advanced Computer Science and Applications (IJACSA)*, vol. 9, no. 1, pp. 18–25, 2018.
- [29] H. Fanai and H. Abbasimehr, "A novel combined approach based on deep autoencoder and deep classifiers for credit card fraud detection," *Expert Systems with Applications*, vol. 217, p. 119562, 2023.
- [30] J. H. Friedman, "Greedy function approximation: A gradient boosting machine," *Annals of Statistics*, vol. 29, no. 5, pp. 1189–1232, 2001.
- [31] T. Hastie and R. Tibshirani, *Generalized Additive Models*. Chapman and Hall/CRC, 1990.
- [32] A. Niculescu-Mizil and R. Caruana, "Predicting good probabilities with supervised learning," in *Proceedings of the 22nd International Conference on Machine Learning (ICML)*, 2005, pp. 625–632.
- [33] L. S. Shapley, "A value for n -person games," *Contributions to the Theory of Games*, vol. 2, no. 28, pp. 307–317, 1953.
- [34] P. Micikevicius, S. Narang, J. Alben *et al.*, "Mixed precision training," 2018, arXiv:1710.03740.
- [35] Vesta Corporation and IEEE Computational Intelligence Society, "IEEE-CIS fraud detection dataset," Kaggle Competition, 2019, available: <https://www.kaggle.com/c/ieee-fraud-detection>.
- [36] B. Goodman and S. Flaxman, "European Union regulations on algorithmic decision-making and a "right to explanation"," *AI Magazine*, vol. 38, no. 3, pp. 50–57, 2017.
- [37] H. He and E. A. Garcia, "Learning from imbalanced data," *IEEE Transactions on Knowledge and Data Engineering*, vol. 21, no. 9, pp. 1263–1284, 2009.
- [38] H. He, Y. Bai, E. A. Garcia, and S. Li, "ADASYN: Adaptive synthetic sampling approach for imbalanced learning," in *Proceedings of the 2008 IEEE International Joint Conference on Neural Networks (IJCNN)*, 2008, pp. 1322–1328.
- [39] C. Elkan, "The foundations of cost-sensitive learning," in *Proceedings of the 17th International Joint Conference on Artificial Intelligence (IJCAI)*, vol. 2, 2001, pp. 973–978.

- [40] B. Yousefimehr and M. Ghatee, "A distribution-preserving method for resampling combined with LightGBM-LSTM for sequence-wise fraud detection in credit card transactions," *Expert Systems with Applications*, vol. 262, p. 125661, 2025.
- [41] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.
- [42] Y. Wu, L. Wang, H. Li, and J. Liu, "A deep learning method of credit card fraud detection based on continuous-coupled neural networks," *Mathematics*, vol. 13, no. 5, p. 819, 2025.
- [43] L. Ding, L. Liu, Y. Wang, P. Shi, and J. Yu, "An AutoEncoder enhanced light gradient boosting machine method for credit card fraud detection," *PeerJ Computer Science*, vol. 10, p. e2323, 2024.
- [44] F. Carcillo, Y.-A. Le Borgne, O. Caelen, Y. Kessaci, F. Oblé, and G. Bontempi, "Combining unsupervised and supervised learning in credit card fraud detection," *Information Sciences*, vol. 557, pp. 317–331, 2021.
- [45] S. Saraf and A. Phakatkar, "Detection of credit card fraud using a hybrid ensemble model," *International Journal of Advanced Computer Science and Applications (IJACSA)*, vol. 13, no. 9, pp. 464–474, 2022.
- [46] A. R. Khalid, N. Owoh, O. Uthmani, M. Ashawa, J. Osamor, and J. Adejoh, "Enhancing credit card fraud detection: An ensemble machine learning approach," *Big Data and Cognitive Computing*, vol. 8, no. 1, p. 6, 2024.
- [47] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.
- [48] C. R. Harris, K. J. Millman *et al.*, "Array programming with NumPy," *Nature*, vol. 585, no. 7825, pp. 357–362, 2020.
- [49] W. McKinney, "Data structures for statistical computing in Python," in *Proceedings of the 9th Python in Science Conference (SciPy 2010)*, 2010, pp. 56–61.
- [50] F. Pedregosa, G. Varoquaux *et al.*, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.
- [51] A. Paszke, S. Gross *et al.*, "PyTorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems 32 (NeurIPS 2019)*, 2019, pp. 8024–8035.
- [52] J. D. Hunter, "Matplotlib: A 2D graphics environment," *Computing in Science & Engineering*, vol. 9, no. 3, pp. 90–95, 2007.
- [53] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, and T.-Y. Liu, "LightGBM: A highly efficient gradient boosting decision tree," in *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 30, 2017, pp. 3146–3154.