TU Dublin, Tallaght Campus

BSc (Honours) in Computing with Artificial Intelligence and Machine Learning (TU862)

Towards Trustworthy AI in Financial Compliance: A Study of Explainable Graph Neural Networks Using Elliptic2

Neri Carcasci

School of Enterprise, Computing and Digital Transformation

Supervised by

Dr. Musfira Jilani
School of Enterprise, Computing and Digital Transformation

April 26, 2026

Declaration

I hereby certify that the material, which I now submit for assessment on the programmes of study leading to the award of Bachelor of Science, is entirely my own work and has not been taken from the work of others except to the extent that such work has been cited and acknowledged within the text of my own work. No portion of the work contained in this thesis has been submitted in support of an application for another degree or qualification to this or any other institution.

_____

Neri Carcasci
April 26, 2026

Acknowledgements

I would like to thank my supervisor, Dr. Musfira Jilani, for her guidance, feedback, and support throughout the duration of this project. I would also like to thank the staff of the School of Enterprise, Computing and Digital Transformation at TU Dublin for the knowledge and skills they have imparted over the course of my studies.

An acknowledgement is also made to the large-scale language model Claude Opus 4.7, which was used to assist in refining the language and academic style of this thesis. The prompt used was: *In the next prompt I will give you a paragraph of text, please correct any English grammar or academic style issues with the text, but do not simplify or paraphrase.* All technical content, analysis, and conclusions are my own.

List of Figures

List of Tables

Contents

Abstract

Regulatory regimes like the EU AI Act and the ALTAI guidelines mandate that anti-money-laundering compliance teams using blockchain transaction data deploy models which are not only accurate but also auditable and interpretable. Graph neural networks are an obvious solution for transaction data; however, the published baselines for the Elliptic2 benchmark do not provide performance metrics for GNNs using node features or the ability to interpret model explanations, raising the questions of what GNNs with features are able to achieve on this task and the extent to which we can interpret their results.

We show how a state-of-the-art GNN, a two-layer Graph Attention Network v2 with Gated Attention Pooling, can be trained on the labelled subgraphs of Elliptic2 without using the 196 million-edge background graph, using only the 43 node features. The primary model achieves a test PR-AUC of 0.515 and a ROC-AUC of 0.934. Against the reported structure-only benchmarks, this represents a 2.47× increase in PR-AUC, but the correct interpretation is not that this work has developed a better architecture, but that the discriminative signal in Elliptic2 is in node attributes rather than macro-topology.

On the explainability of these models, we show that Integrated Gradients and Kernel SHAP agree on the global feature ranking (Spearman rank correlation of 0.95), and that the discriminative features are revealed in a feature signedness contrast as opposed to in their attribution magnitude. GNNExplainer reveals explanatory features that are consistent with the global feature rankings for all 200 stratified test graphs, whereas the PGExplainer algorithm fails to produce informative masks under four training conditions, which we attribute to a known issue which must be addressed in future work.

1. Introduction

Every blockchain stores all the transactions it has ever carried out, and tens of billions of dollars are being laundered on blockchains each year (Chainalysis, 2024). Compliance officers are required to monitor this activity under anti-money-laundering (AML) law, and since manually checking blockchain transactions at this scale is not possible, they have adopted machine learning. The European Union (EU) AI Act, which went into effect in 2024 (European Parliament and Council of the European Union, 2024), labels AML as a high-risk application under Annex III §5(b), and Articles 9, 13, 14, and 15 attach binding obligations on risk management, transparency, human oversight, and robustness. The Assessment List for Trustworthy Artificial Intelligence (ALTAI) (High-Level Expert Group on Artificial Intelligence, 2020) further describes seven requirements for the deployment of trustworthy AI. Without explainability, auditability, and stress-testing, an AML model is no longer viable from a regulatory point of view.

The natural choice of models for graph-structured transaction data are graph neural networks (GNNs). The Elliptic2 dataset (Bellei et al., 2024), used as the primary dataset in this thesis, is the largest fully labelled public AML dataset, comprising 121,810 labelled subgraphs on a 196M-edge background graph, with 2.27% positives. The Elliptic2 dataset comes with a number of baselines that evaluate different structural graph models on AML tasks (the GLASS labelling trick from Wang and Zhang (2022), GIN and GraphSAGE) without using node features, achieving a maximum PR-AUC of 0.208. These baselines do not assess the explainability of their models, which is what this thesis explores.

We therefore seek to answer the following question: Can an explainable GNN trained on node features only, on the labelled subgraphs of the Elliptic2 dataset, match or exceed structure-based baselines on the AML task, and do the resulting explanations yield meaningful insights for analysts?

We make four contributions: (i) a feature-aware subgraph-local benchmark on AML task performance, on which we report that a two-layer GATv2 model with global attention pooling achieves 0.515 PR-AUC on the held-out test set, approximately a 2.47× increase over the best structure-only baseline, with the regime caveat that the comparison locates the signal in node attributes rather than ranking architectures; (ii) our evaluation of Integrated Gradients and Kernel SHAP on the trained model, with rank-correlation reported across the leading discriminative features; (iii) our broader-than-typical evaluation of structural explainers (GNNExplainer, GATv2 attention, SubgraphX), with mask entropy proposed as a fidelity diagnostic on the median three-node subgraph where standard fidelity fails; (iv) our mapping of the pipeline against ALTAI and the EU AI Act, identifying the gaps in current GNN explainability that prevent compliance-grade deployment.

2. Literature Review

2.1 Scope: anti-money-laundering on the blockchain, and why graphs

Money laundering is not a niche compliance problem: the United Nations Office on Drugs and Crime estimates as much as USD 2 trillion of value is laundered worldwide annually, a number that the Financial Action Task Force repeats verbatim in its most recent Annual Report (Financial Action Task Force, 2025; United Nations Office on Drugs and Crime, 2024). The same FATF report finds that fraud is the reported predicate offence in 89% of mutual-evaluation reports from the 156 jurisdictions it has evaluated, ranking second only to corruption. These two statistics frame the problem space in which we operate: large, non-trivial flows of value on behalf of criminals are moving through the financial system, and the laundering that ensues is a multi-actor, connected process, not a single isolated transaction.

This connectedness is precisely what makes graph methods so well-suited to the problem. In typical laundering schemes, criminals will chain transactions and intermediaries to try to evade detection, because their schemes are built in such a way that no single transaction can be caught by a transaction-by-transaction filter. Bellei et al. (Bellei et al., 2024) formalise this by doing the labelling at the connected component level (i.e., the whole laundering scheme), not the transaction level; we follow this definition throughout this thesis.

The blockchain makes for a particularly good substrate on which to explore these techniques. Bitcoin is pseudonymous, not anonymous: every transaction can be seen by anyone, every transaction input can in theory be traced, and yet it is hard to connect transactions to real-world entities. It is exactly this hole in the system that the commercial transaction intelligence companies Elliptic, Chainalysis, and TRM Labs have built billion-dollar businesses on, but is also a fertile field for new academic research given that the complete transaction graph underlying Bitcoin is permissionless, and can be reconstructed from scratch from any blockchain node. The first to release a large-scale subset of this with labels was Weber et al. (Weber et al., 2019), and in the intervening years, work has evolved from basic rule-based heuristics to classical ML baselines (Random Forest, XGBoost) to the kinds of graph-native ML models we are primarily interested in today. The rule-based, non-graph era will be of interest only in providing historical background; the comparison we are interested in today is between graph methods, and also between explainable vs. opaque variants of those graph methods.

2.2 The Elliptic family and the labelled-subgraph regime

There is now a growing set of Bitcoin AML datasets that can be used to build and benchmark graph methods. The lineage begins with Elliptic1 (Weber et al., 2019), a subset of 203,769 transaction nodes and 234,355 directed edges, with 49 temporal snapshots in

which each transaction is either labelled illicit, licit, or unknown. Elliptic1 is real and well-documented, and is an important milestone for the AML graph ML community, but its label is at the level of a single transaction, so the graph structure gets collapsed into a per-row tabular problem as we described above. In Elliptic++ (Elmougy and Liu, 2023), Elmougy and Khalil add an additional node type of Bitcoin address, leading to a more natural graph structure, but they don't offer any new labels or ground truth (their labels remain at the level of the single node, i.e., single transaction).

In contrast, AMLSim (Suzumura and Kanezashi, 2018) and IBM's newer AMLworld (Altman et al., 2023) have taken the opposite approach of generating synthetic transaction streams where the underlying AML schemas are known as part of their generation process. Synthetic data is an attractive alternative for benchmarking explainability methods because the "correct answer" will always be well-defined, so there is no need to guess what the answer ought to be if the "right" model is selected to evaluate the explanation quality of the model. On the other hand, however, any explanation quality claims based on synthetic data are ultimately claims about the synthetic data generator, not real-world laundering activity. In Chapter 3, we will talk a little more about how we decide between synthetic data and real-world data in motivating our final dataset choice. In 2024, MIT, IBM, and Elliptic jointly released Elliptic2 (Bellei et al., 2024), a dataset that effectively launches the labelled-subgraph research space at large scale. Comprising 121,810 connected components within a background graph of approximately 49.4M nodes and 196.2M edges, the Elliptic2 dataset has one single licit-or-suspicious label for every subgraph, given by Elliptic's investigation team. This means that the positive rate of Elliptic2 is 2.27% with an imbalance ratio of 43:1. The unit of classification for this dataset was deliberately selected to match the decision unit of AML analysts when generating a Suspicious Activity Report, which is why it can be considered a suitable substrate for answering our research question.

2.3 GNN architectures for subgraph classification

A graph neural network updates node representations through iterative neighbourhood message passing. The general message passing update is formulated by Gilmer et al. (Gilmer et al., 2017), where the hidden state of each node is updated by aggregating the features of its neighbours and then combining the aggregated neighbour features with the hidden features of the node itself using a learnable function. In our evaluation, these architectures primarily vary in the parameterisation of the aggregation function. The graph convolutional network (GCN) (Kipf and Welling, 2017) learns node representations through neighbourhood convolution with adjacency-weighted average of neighbour features. GCNs are the simplest and most computationally-efficient baseline architecture but limit their capacity by assigning equal weights to neighbouring nodes after applying a symmetrical normalisation to the adjacency matrix. GraphSAGE (Hamilton et al., 2017) improves GCNs by sampling a fixed number of neighbours of each node and parameterising the aggregation function (mean/LSTM/Pool), which also induces an inductive bias that allows generalisation to subgraphs that were not part of the training data. This inductiveness property of GraphSAGE is particularly important for operational relevance because our target task is subgraph classification of subgraphs that were not observed in training data. Graph Attention Networks (GAT) (Veličković et al., 2018) further enhance

the capacity of GCNs and GraphSAGE architectures by incorporating learned attention coefficients between a query node and all neighbours during message passing. Brody et al. (Brody et al., 2022) note that GATs are in fact static attentions, and propose a variant they call GATv2 where the order of computations is swapped, which results in a dynamic attention scheme where node-dependent attentions are learned during message passing. We adopt GATv2 in our evaluation for two reasons: (1) the dynamic attention scheme provides GATv2s with more expressive power over GATs; and (2) edge weights in the GATv2 model provide an interpretable output that allows us to further analyse the model in Chapter 6. Finally, we considered NNConv (Gilmer et al., 2017) for its edge-features-aware message passing. NNConv learns a set of independent convolution parameters for every dimension of the edge feature vector and the hidden dimension. Specifically, its number of learnable parameters has a cubic dependency with respect to the width of the hidden vector and a linear dependency in the edge feature dimension ($\mathcal{O}(d_e \cdot d^2)$). This parameter efficiency does not make NNConv a candidate architecture for subgraphs with small numbers of nodes; in fact, we show in our pre-registration study in Chapter 5 how this is the key reason why we terminate NNConv during the architecture search in the beginning. A final important design decision for graph-level classification is the pooling operation. Since graphs in our dataset vary in size from 2 to 296 nodes, the use of mean or max pooling for graph-level classification is not desirable, because of the loss of information about node attributes induced by both of these methods. Gated pooling, proposed by Li et al. (Li et al., 2016), learns per-node importance scores that are then aggregated to form a node representation, which is why we choose gated pooling as the pooling operator for our models. Finally, the Jumping Knowledge concatenation method proposed by Xu et al. (Xu et al., 2018) is adopted during our architecture search to avoid over-smoothing in deeper model stacks.

In conclusion, we also consider Wang and Zhang's label-trick approach (Wang and Zhang, 2022). Their study shows that by feeding an indicator of the input subgraphs (i.e., their GLASS framework), subgraph tasks recover the expressive power missing in standard message-passing. This provides the theoretical underpinning for using the labelled-subgraph indicator as a learnable feature instead of as a topological constraint, we adopt their result to use the indicator from here on.

2.4 Explainability for GNNs

Explainability for GNNs falls into two buckets which address distinct questions, fulfil distinct criteria, and impose distinct costs; we discuss each.

2.4.1 Feature attribution

The first bucket is based on the observation that a GNN ultimately takes in a feature matrix, so any feature attribution method designed for standard tabular data can be ported directly to it. LIME (Ribeiro et al., 2016) starts here: given an instance of interest, it perturbs the feature matrix, fits a simple linear sparse model on the neighbourhood, and returns the linear coefficients as the feature attributions. SHAP (Lundberg and Lee, 2017) unifies LIME and various other methods in the additive-feature-attribution paradigm, and proves that the only method meeting the local accuracy, missingness, and

consistency axioms is the Shapley value from cooperative game theory. Sundararajan et al. (Sundararajan et al., 2017) instead introduce Integrated Gradients, which integrates the model's input space gradients over a straight-line interpolation from a reference to the instance under study, and which satisfies sensitivity and implementation invariance. All three methods have rigorous axiomatic underpinnings, and all three apply to GNNs by regarding the node feature vector as the unit of explanation. This is useful when one wants feature-dictionary interpretability, which is relevant to our question of which of Elliptic2's 43 anonymised node features hold the distinguishing power. The downside is that no feature attribution method can see the graph: a feature attributed value by SHAP tells us nothing about which nodes were involved in the explanation or how they were connected.

Given these trade-offs, we treat the feature attribution methods as our baseline diagnostic (Ch. 6), reporting that Integrated Gradients and Kernel SHAP correlate at $\rho = 0.9505$ on the global feature ranking. The agreement of two attribution methods based on different assumptions is necessary but not sufficient to ensure trustworthy results; while it remains possible that both agree for the wrong reasons, the probability of them agreeing purely by coincidence is much lower.

2.4.2 GNN-native explainers

The second bucket treats the structural contribution to the prediction as a primary object. GNNExplainer (Ying et al., 2019) frames explanation as a soft mask applied to the input edges and nodes, optimised to maximise the mutual information between that masked subgraph and the model's prediction. By employing a mean-field variational approximation, it enables optimisation for individual instances; however, this yields explanations that are localised to a specific node and not immediately generalisable. SubgraphX (Yuan et al., 2021), on the other hand, approaches the task as the identification of a discrete subgraph whose predictor output best approximates the initial one. This involves running a Monte Carlo Tree Search algorithm guided by a Shapley-value measure limited to the node's L-hop neighbourhood, thereby linking back to the feature-level SHAP lineage. SubgraphX has a high computational cost when compared with GNNExplainer as a result of the Monte Carlo Tree Search step, and in Chapter 6 we, thus, only present it over a stratified sample of 50 subgraphs. Lucic et al. (Lucic et al., 2022) take explainability further into a counterfactual: they remove edges from the adjacency matrix until the model's prediction is altered, returning a minimal subgraph that is just sufficient to do so. This is a kind of recourse explanation rather than of justification. As with SubgraphX, we will discuss the position of counterfactual GNN explanations with the pipeline in Chapter 6. A common thread among them that needs to be stressed here is that they are not more explanatory than feature attribution per se: rather than being more, they simply are more about something else, which is to say that they identify a subgraph in which the prediction was made, whereas attribution identifies the features that drove that subgraph to make the prediction. These are two families of explanations answering two different questions, and we maintain throughout this work that any production-grade compliance system will need both.

2.5 Trustworthy AI as a testable hypothesis

The literature on AI ethics is extensive; the literature on AI ethics for AML, in particular, is scant. The European Commission's High-Level Expert Group on AI published the Assessment List for Trustworthy AI (ALTAI) in 2020 (High-Level Expert Group on Artificial Intelligence, 2020), which distilled the broader Ethics Guidelines into 7 dimensions: Human agency and oversight; Technical robustness and safety; Privacy and data governance; Transparency; Diversity, non-discrimination and fairness; Societal and environmental well-being; and Accountability. These 7 dimensions are to be operationalised not as a list of requirements to be checked, but as a hypothesis to be tested with the pipeline we construct. Our hypothesis, then, is this: if the best performing graph neural network on Elliptic2 that uses attention-based pooling and is equipped with a GNN-native explainer is also able to satisfy the 7 dimensions of ALTAI, then the evidence for deploying graph methods in a compliance-grade manner is bolstered; otherwise, the pipeline will at least expose which dimension(s) the current state-of-the-art in explainability fail(s) to support, which in itself is a contribution to the field of trustworthy AI.

This is not a random collection of constraints. The EU AI Act (Regulation 2024/1689 (European Parliament and Council of the European Union, 2024)) came into effect in 2024 and obliges high-risk AI system providers and deployers to comply with a defined set of obligations. These include ongoing risk management (Article 9), data and data governance standards (Article 10), a record-keeping obligation spanning the full system lifespan (Article 12), ensuring transparency for users (Article 13), human oversight including the mitigation of automation bias (Article 14), and ensuring accuracy, robustness, and cybersecurity (Article 15). Moreover, any individual concerned with a decision taken by a model listed in Annex III has the right to receive an effective and meaningful explanation of the reasoning behind the decision (Article 86). Annex III itself excludes pure financial-fraud detection from the high-risk list in point 5(b). However, the exclusion may be somewhat superficial because an AML model which ultimately gates access to services is on the borderline of a creditworthiness determination. As we argue in Chapter 7, this has relevance. These same requirements also align with the FATF's risk-based approach (Financial Action Task Force, 2025). Recommendation 1 requires regulated entities to allocate resources based on risk, which is the organisational analogue to our technical assertion that an explainable triage model provides higher value than an opaque model which prioritises high recall. We quote the following directly from ALTAI, footnote 20: "accuracy is not enough; we need other metrics for the imbalanced case as well... For a model with multiple classes, the F1-score and the per-class FPRs and FNRs are a valid set of measurement metrics for evaluating model performance." This is precisely the set of metrics recommended by the EU expert group, the very one that we advocate for against single metric benchmark claims. This section makes clear that none of the currently available GNN architectures or explanation methods fully covers all the requirements of the 7 ALTAI pillars. For example, robustness per Article 15 demands resilience to data poisoning and adversarial examples (see (Jin et al., 2021)), which are only now being studied in the GNN field. In turn, transparency per Article 86 calls for explanations that a non-expert user can take action on, which is a much stricter definition than many of those in the explainability field. Likewise, diversity and non-discrimination cannot be measured on a feature-anonymised dataset like Elliptic2 using the usual fairness metrics. These difficulties are the core problem that will be addressed by our ethical assessment

in Chapter 7.

2.6 Synthesis: the pieces and how they fit

Combining all five sections above, we are left with our design space. We identified the problem to be real and evolving (2.1). We chose Elliptic2 (2.2) to be the substrate through which we study it. GATv2s with attention pooling (2.3) are the appropriate model class for this substrate. The explainability methods that should be assessed jointly are a feature-attribution baseline (Integrated Gradients and Kernel SHAP) and a GNN-native set of methods (GNNExplainer and SubgraphX) with CF-GNNExplainer as the counterfactual comparison (2.4). The trustworthy-AI hypothesis we are testing is whether this assembled pipeline can satisfy each of ALTAI's seven dimensions, in light of the EU AI Act's binding requirements and the FATF risk-based framing (2.5). What this synthesis makes visible is a chain of assumptions. We assume that the labelled-subgraph regime is the correct grain at which to study AML; that the dynamic-attention property of GATv2 is worth the additional implementation complexity over GraphSAGE; that the convergence of two different attribution methods is itself evidence about the model rather than about the methods; that a GNN-native explainer can produce subgraph-level evidence that a deployer can use. Each of these assumptions is defensible and each is open to challenge, and we revisit each in turn in the empirical chapters that follow.

2.7 Positioning and gap statement

Three gaps in the published literature define the contribution of this thesis. First, no published work on Elliptic2 has reported a subgraph-classification result that uses the node features rather than only the topological structure of the labelled subgraphs. Bellei et al.'s original baseline relies on the GLASS labelling trick over the background graph and does not exploit the per-node attribute vectors that Elliptic2 ships with; we report that a feature-aware GATv2 with attention pooling exceeds this structure-only baseline by a factor of approximately 2.4 in PR-AUC on the labelled-subgraph universe, which is interesting precisely because it suggests that the discriminative signal is concentrated in the node attributes rather than in the macro-topology. We are careful in Chapters 6 and 9 to caveat the regime in which this multiplier applies. Second, the empirical evaluation of structural explainers on small AML subgraphs is thin. Most published evaluations of GNNExplainer and SubgraphX use citation networks or molecular graphs, where component sizes are larger and topological motifs are richer than in the Elliptic2 distribution (a median of three nodes per component, with a long right tail). We report fidelity, sparsity and stability metrics on this regime in Chapter 6 and discuss the limits of fidelity as a measurement when the input graph is barely larger than a single edge. Third, the existing literature treats the operating-point question (where on the precision-recall curve a compliance team should sit) largely as a model-tuning detail rather than a workflow-design choice. We argue, drawing on FATF's risk-based framing, that operating-point selection is itself a trustworthy-AI question because it determines the false-positive burden a human reviewer must absorb, and we present operating-point analysis as a first-class contribution in Chapter 6. These three gaps motivate the central research question

that organises the rest of the thesis: can an explainable graph neural network trained only on the node features of labelled Elliptic2 subgraphs match or exceed structure-based baselines on the AML task, and do the resulting explanations provide actionable signal for compliance analysts? The chapters that follow take this question apart piece by piece.

3. Dataset Selection

This work uses the Elliptic2 dataset (Bellei et al., 2024), a public benchmark of labelled Bitcoin subgraphs jointly released by MIT CSAIL, the MIT-IBM Watson AI Lab, and Elliptic. We chose Elliptic2 purposefully, after evaluating other realistic options for graph-based AML, because it is the only one meeting all three of this thesis's conditions: subgraph-level labels, non-synthetic blockchain transactions, and a scale at which the explainability claim is meaningful.

3.1 Options considered

The five realistically viable AML GNN thesis datasets are Elliptic1 (Weber et al., 2019), Elliptic++ (Elmougy and Liu, 2023), AMLSim (Suzumura and Kanezashi, 2018), the IBM "AMLworld" benchmark (Altman et al., 2023), and Elliptic2 (Bellei et al., 2024). Each dataset was considered against four dimensions pertinent to this thesis' objectives: whether they comprise real or synthetic transaction data, their labelling granularity (node-level vs. subgraph-level labelling), size (large enough that results generalise, yet manageable on a student computer), and license (open enough to reproduce findings and release code).

Elliptic1 (Weber et al., 2019) was the first widely-adopted Bitcoin AML benchmark, containing 203,769 transaction nodes, 234,355 edges in a directed manner, and 49 discrete temporal snapshots. Although it consists of real data, has a permissive license, and is well-documented, each node (transaction) is labelled; furthermore, it assumes each transaction is either licit or illicit. This violates both this thesis' underlying premise that AML is a property of related transactions in the graph, and the need for subgraph-level labelling to support a thesis whose accountability claim (Chapter 7) rests on identifying suspicious entities via AML GNN.

Elliptic++ (Elmougy and Liu, 2023) expands the transaction graph by adding Bitcoin addresses as nodes, while retaining node-level labelling of both addresses and transactions. This creates a richer relational data structure, but no new evaluation approach or source of ground-truth is provided. The dataset was considered as a supplementary dataset against which to validate the Elliptic2 results, but as no new ground-truth was added, Elliptic++ was not a viable primary dataset for this thesis.

AMLSim (Suzumura and Kanezashi, 2018) is an artificial AML transaction generator from IBM that provides full control over the amount of illicit activity present, as well as typology, volume, etc. AMLSim provides a perfect ground-truth by construction; however, that ground-truth is merely that which was told AMLSim to provide. In other words, an explainability claim for a model trained on AMLSim is only that the model can learn to identify a particular pattern the simulator was instructed to simulate, the same as any other model trained and run on AMLSim.

IBM's "AMLworld" dataset (Altman et al., 2023) is a more recent, sophisticated artificial benchmark dataset that incorporates agent behaviour, several typologies, and temporal dynamics into a more realistic environment than AMLSim. While it is more realistic, it is still artificial and only released after this thesis' data selection was decided.

Finally, we consider Elliptic2 (Bellei et al., 2024). This is the only dataset from among these options that is both real and labelled at the subgraph level, at scale.

3.2 Rationale for Elliptic2

This thesis uses Elliptic2 for three reasons. First, its unit of labelling matches the research question. Each of Elliptic2's 121,810 labelled subgraphs maps 1:1 to a connected component of Bitcoin entity transactions and transactions in the dataset, and has been assigned a single licit/suspicious status by Elliptic's investigations team. The unit at which an AML compliance team makes a decision, which results in the filing of a Suspicious Activity Report, is a component of linked transactions rather than any single transaction. Consequently, the subgraph-level granularity of the data corresponds to the operational reality AML GNNs must ultimately explain. It also represents the appropriate granularity at which to evaluate a graph-level GNN end-to-end.

Second, it is large enough that a study can be meaningfully scaled. The data is drawn from a background graph of approximately 49.4M entity nodes and 196.2M transaction edges. The fact that the labelled entities in Elliptic2 are in turn drawn from such a large universe ensures that any model, as well as any explainability algorithm or strategy, evaluated on this data is evaluated at a volume consistent with the magnitude of data a real transaction monitoring system must face. A study evaluated on a few thousand synthetic transactions simply wouldn't have the same evidentiary value.

Third, it is real. The background graph is extracted directly from the Bitcoin blockchain; and the subgraph labels are based on commercial classification data, rather than an artificial generator seed. It is true that using real data means ground-truth isn't perfect, as we address below; however, at least we can assert that any behaviour the model identifies as suspicious is at the very least a pattern of suspicious activity that occurs in the real world, not just a pattern a synthetic generator created.

A secondary but nonetheless relevant reason we chose Elliptic2 is that it is released under the Apache-2.0 licence, which allows reproduction, redistribution, derivative works, and public distribution of any work released under that licence. This allows us to demonstrate reproducible experiments (Chapter 7), a key aspect of the accountability claim on which this thesis rests.

3.3 Dataset properties and limitations

In this dataset's labelled universe, we find that the 121,810 components have 119,047 labelled as licit and 2,763 labelled as suspicious. This gives a positive rate of 2.27% and negative/positive ratio of 43/1. Each node has 43 ordinal features, and each edge has 95 ordinal features, all anonymised and pre-binned by the dataset authors. The five primary CSVs, `background_nodes.csv`, `background_edges.csv`, `nodes.csv`, `edges.csv`, `connected_components.csv`, total approximately 90 GB on disk. These will be used in the data pipeline discussed in Chapter 4.

However, the data's limitations must also be noted, because they impact how the following results can be interpreted: the dataset features have all been anonymised. This

was a design choice made by Elliptic, but it creates an upper bound on explainability. We revisit this issue in Chapter 7. We can see what features signal what, but cannot map those features to their original meanings unless one also has the original feature dictionary, which is not part of the public release.

The data labels are commercial classifications of entities (e.g. addresses, transactions), not adjudicated regulatory ground truth. A suspicious subgraph represents Elliptic's belief that entities in the subgraph were involved in money laundering. This is based on the same investigative process that it sells to its financial clients. This represents the best data labels available for public datasets at this scale. They are a more realistic representation than fake or artificial labels; however, this does not mean that the ground truth has been verified by any regulator to represent real illicit activity. A model learned from these labels will also learn from any biases in the original labelling priors.

Finally, there are no explicit features in the dataset that can be considered protected (demographic, geographic, or institutional), so traditional fairness metrics cannot be applied. Chapter 7 details the issue and introduces the structural bias proxies that were substituted in this thesis. A final structural constraint to keep in mind is that this is a single-chain dataset (Bitcoin only) and that there is no temporal information available in the data. These constraints impact generalisability that can be claimed about any model developed using this data.

4. Data Pre-processing

Elliptic2 comes pre-packaged as three separate tables and a significantly larger background graph. Converting this raw data into graph neural network (GNN) compatible objects is not just a formatting issue; it is where the most significant choices lie. These decisions define the learning grain size, determine the extent to which the data splitting protocol is free of leakage, limit the size of the label universe, and dictate the maximum interpretability the final model can possess. This chapter describes those choices along with the supporting evidence, beginning with the raw files and concluding with the PyTorch Geometric (PyG) objects that are passed to the GNN training routine in Chapter 5.

4.1 Pipeline overview

The entire preprocessing routine is written in two notebooks that run end-to-end autonomously:

- `wb01_e2_preprocessing.ipynb` is dedicated to extracting the label universe, aligning node features, constructing the splits, and generating all reproducibility artefacts.
- `wb03c1_e2_preprocessing_edge_feat.ipynb` augments this pipeline with edge features derived from the 49.3 million node background graph.

Figure 4.1 provides a broad overview of the whole pipeline, beginning with the raw `.csv` inputs, passing through intermediate parquet cache files and packed `.npy` arrays, and ultimately producing PyG `Data` objects consumed during training.

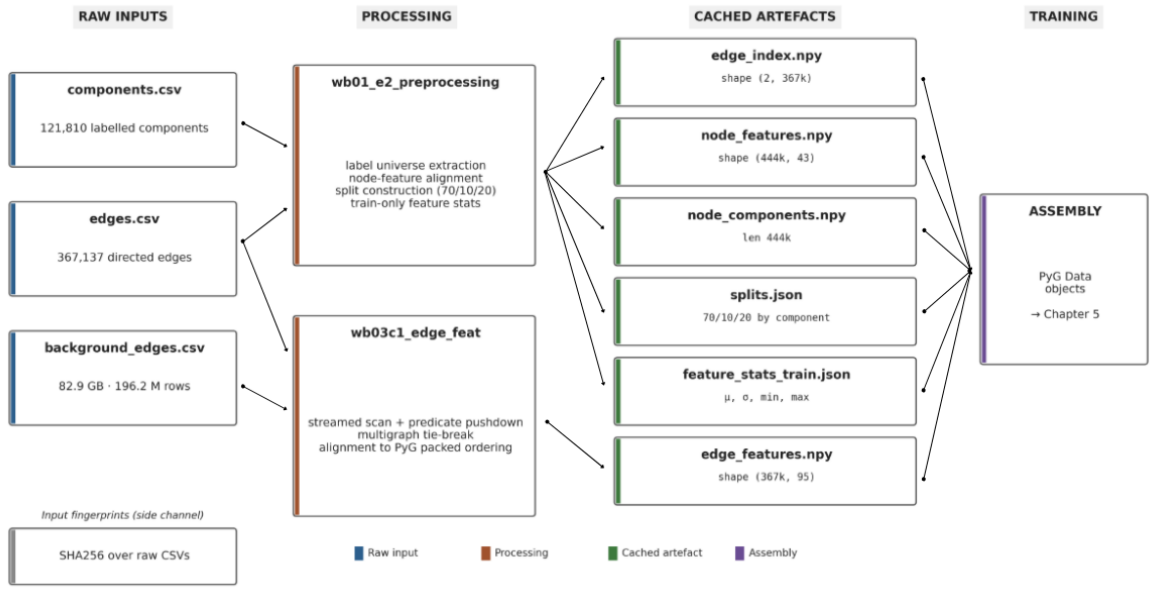

Figure 4.1: End-to-end preprocessing pipeline. The five raw Elliptic2 `.csv` inputs (left) are consumed by two notebooks (middle): `wb01_e2_preprocessing` extracts the label universe, aligns node features, and constructs the 70/10/20 splits with train-only feature statistics; `wb03c1_e2_preprocessing_edge_feat` streams the 196.2 M-row background edge file with predicate pushdown, applies multigraph tie-breaking, and aligns edges to the PyG packed ordering. The six cached artefacts (right) are loaded by every downstream notebook as immutable inputs and assembled into the PyG `Data` objects consumed in Chapter 5. SHA256 fingerprints over the raw inputs are written at the `wb01` boundary as a reproducibility side channel.

The output of each notebook is cached as an intermediate artefact that can be reloaded by other notebooks and treated as immutable read-only inputs at that point. The full inventory of intermediate artefacts appears in Section 4.6. If any metric later in the pipeline fails to replicate the results presented here, the first step to diagnose this would be to recompute the input fingerprints (SHA256 hashes over the input `.csv` files) and compare against a fresh git checkout of the same dataset. If these input hashes match between the two runs, we can conclude that the inputs are identical and the discrepancy is located elsewhere within the analysis.

4.2 Graph construction

The label universe is defined by all nodes whose connected component `ccLabel` falls into either of the "licit" or "suspicious" classes. Any node that is not part of the label universe belongs to the background graph and will not be considered as part of our GNN training or testing in this thesis. Excluding all background graph nodes from training/evaluation is a scope choice, not a technical constraint. Future work can extend to background graph nodes, as we revisit in Chapter 9.

Having extracted the graph of all licit and suspicious components, this graph consists of 444,521 nodes, 367,137 directed edges, and 121,810 connected components. We run the

following three sanity checks before generating any tensors on disk:

1. **Non-null checks:** Every column of type `clId` and `ccLabel` is verified to be entirely free of null values. All identifiers are explicitly cast to 64-bit integers.
2. **Referential integrity:** We verify that the `edge_index` columns reference a known, unique labelled node.
3. **Node completeness:** We verify that every component `ccId` is represented as having at least one node `clId`.

After these checks, we record the following values:

- `unknown_src`: 0
- `unknown_dst`: 0
- number of components labelled but never referenced: 0
- number of nodes with `ccId` unregistered in components: 0

The raw `clId` attribute is a high-cardinality value with non-negligible sparsity, meaning the `clId` column is neither a suitable index for tensors nor directly usable as node indices in an adjacency list. Instead, all `clId` values are transformed to compact, contiguous integer representations. Consequently, after running the transformation described above, the shape of `edge_index` is $(2, 367{,}137)$ and the length of the `node_components` column is 444,521, both of type `int64` on disk. The connected components range in size from a minimum of 2 nodes up to a maximum of 296 nodes, with an average component size of 3.649 nodes. These values imply a strongly right-skewed distribution over component sizes, with most connected components consisting of a very small number of nodes. This property of the dataset influences a number of downstream decisions, including the choice of pooling operator in Chapter 5, which must be capable of operating over a graph whose underlying components are predominantly very small (often just a few nodes), as well as the subgroup analysis discussed in Chapter 6.

4.3 Feature treatment

The handling of the node- and edge-wise feature vectors differs substantially because Elliptic made them available to us in distinct forms.

Node features

The 43 node-wise features are all binned ordinal integers, with the number of unique values ranging from 3 to 99 values, depending on the feature. The authors of the Elliptic2 dataset note that, prior to publication, these originally continuous data attributes such as cluster size, transaction count and so on were binned to protect their intellectual property (Bellei et al., 2024). Because the binning was an explicit choice by the dataset authors to encode

the ordinal nature of the feature space rather than discarding the structure, we retain the integer features without converting them to standardised values. The integer values are all of type `int8`. If there are any missing values in this table, they are imputed to a sentinel value of 0. After performing the semi-join between the edges and labelled universe, there are no missing values in these 43 feature columns for any row. Under the current data splits, the feature values for rows in the train, validation, and test partitions are always complete, so the imputation of 0 does not perform any real imputation operation and is instead used as a safety measure.

We compute the summary statistics for these 43 features using only the training data, which are then stored in a file called `feature_stats_train.json`. We compute these train-only summary statistics in order to prevent any data from the validation or test sets from leaking into later normalisation/scaling experiments; for example, we must ensure that when training the GNN with a standardisation operation, we do not accidentally train on statistics that were computed from the entire labelled universe, including the test nodes. For all 43 feature values, we see that: Feature 19 is mean 18.97, maximum value 98. Feature 23 is mean 18.52, standard deviation 27.17. Across all feature columns, the minimum and maximum feature values are 0 and 98, respectively.

Edge features

Edge features present a more complex challenge. The background edge file consumes 82.9 GB of disk space and comprises 196,215,606 rows. Of these, only 367,137 rows correspond to edges present in the labelled universe. We retrieve those rows by performing a streamed scan with predicate pushdown, taking roughly 679 seconds on our development machine, and then we align them to the packed edge ordering that PyG utilises.

It is important to get multigraph diagnostics correct, because aligning edge-feature rows to the packed edge ordering fails silently if any endpoint pair is not unique and a fallback option is not provided. The 367,137 directed edges correspond to 343,192 unique `(clId1, clId2)` pairs, so there are 23,945 (or 6.52%) edges that have an endpoint pair matching another edge. There are another 14,672 edges that have a reverse exact match in the pair universe. A deterministic fallback breaking ties in edge position in the PyG-packed `edge_index` resolves these, and there are zero edges that remain unmatched following alignment. We sampled 10 random index pairs and re-derived the indices via a raw background scan; in all cases the i-th row of `edge_features.npy` aligns with the i-th edge of `edge_index.npy`, which satisfies our invariant for all downstream computation.

All 95 edge features are continuous in value with a range $[-1, 98]$, a mean of 16.94 and a standard deviation of 23.32. Zero-rates are not consistent across features. The mean zero-rate for all 95 features is 44.02%. Setting a zero-rate threshold of 0.5 provides a dense set of 52 features with a mean of 29.11 and a standard deviation of 24.40. The remaining 43 high-sparsity features are kept for the full array but discarded from the dense version used in the edge feature experiments reported in Chapter 5. Metadata regarding these dense subset column membership is kept on file, `edge_feature_dense_meta.json`, rather than embedded in source code.

Figure 4.2 summarises the 95 edge features along four diagnostic axes: cardinality, sparsity, central tendency, and pairwise correlation among the first 20 features. The cardinality panel shows that most features take fewer than a thousand unique values, with

a small set of high-cardinality channels reaching the tens of thousands. The zero-rate panel motivates the dense-subset cut: a band of features sits above 0.7 zero-rate, dominated by the heavy zero-mass that an edge-aware model would have to learn through. The mean panel reveals that the surviving dense features are also the ones with non-negligible mean magnitude. The correlation block shows clusters of strongly co-moving channels among the first 20 features, indicating substantial information redundancy in the raw 95-dimensional space.

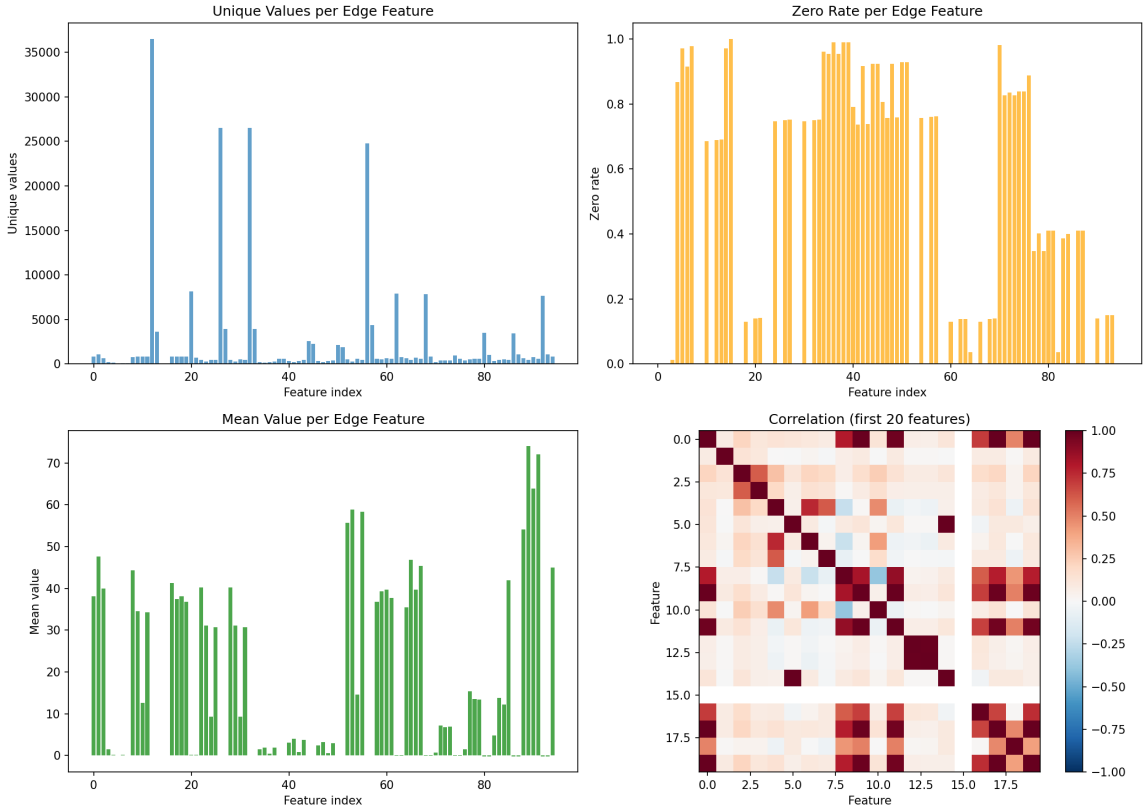

Figure 4.2: Edge-feature diagnostics on the 367,137 directed edges of the labelled-universe slice. Top-left: number of unique values per feature. Top-right: zero-rate per feature, with the 0.5 threshold separating the dense subset of 52 features used in Chapter 5 edge-feature experiments. Bottom-left: per-feature mean. Bottom-right: pairwise correlation among the first 20 features.

4.4 Splitting strategy

The split is performed at connected component level, and never at node or edge level. The alternative, random node split, assigns nodes from the same subgraph into different parts, causing information leakage from the training part to the test part. This is not negligible: on graphs with the median length three, a single leaking neighbour can render a test node prediction nearly trivial. No subsequent modelling choice is capable of repairing this leakage, and for that reason the node-level results on Elliptic1 (Weber et al., 2019) (that involve classifying individual transactions rather than subgraphs) are not strictly comparable to this component level classification: the classification granularities

differ as well as leakage risks. We split component IDs using a two-stage stratified procedure: the first stage holds out 20% of the components for testing, and the second stage splits the remaining 80% into training and validation sets with a 7:1 ratio. This results in 85,267 training components, 12,181 validation components, and 24,362 test components (approximately 70/10/20). Stratification holds the positive class prevalence nearly constant: train, validation, and test splits have positive rates of 2.2682%, 2.2658%, and 2.2699% respectively, with negative-to-positive ratios of 43.09, 43.13, and 43.05. All three splits are serialized into JSON, and we use them unmodified throughout the experiments in this thesis.

4.5 Class imbalance handling

With 43 negatives per positive, a classifier that just predicts negative every time will achieve 97.7% accuracy and zero positives. This is the rationale behind the PR-AUC-first metric in Chapter 5, but it also motivates a training-time fix to the loss function. We use inverse-frequency class weighting with binary cross-entropy loss:

$$w_c = \frac{N}{2\,N_c} \tag{4.1}$$

where N is the number of training components and N_c is the count for class c. This gives a weight of 43.1 for the suspicious class and 1.0 for the licit class. We chose this method over the usual alternatives of focal loss and over-sampling. SMOTE (Chawla et al., 2002) constructs feature values between minority-class samples in feature space, a technique meant for data with only tabular features; applied to component graphs, it would synthesise feature values for non-existent graph structures. The difficulty of synthesising plausible graph structures to match these feature vectors (a problem that is not currently solved) would confuse both the trained graph neural networks and the explainers used in Chapter 6. Focal loss (Lin et al., 2017) changes the loss landscape in ways that would interact badly with the sweep over validation thresholds used in Chapter 5 to find an F1-optimal cutoff for the metric. Class weighting by class frequency, on the other hand, is simpler and more directly interpretable, while maintaining the training data distribution unaltered.

4.6 Reproducibility artefacts

Every result reported later in this thesis is derived from these six files created in this chapter:

- `edge_index.npy` (shape $(2, 367{,}137)$, dtype `int64`)
- `node_features.npy` (shape $(444{,}521, 43)$, dtype `int8`)
- `edge_features.npy` (shape $(367{,}137, 95)$, dtype `float32`)
- `node_components.npy` (length 444,521, dtype `int64`)

- `splits.json` (training/validation/test sets, a JSON map from split name to set of component-IDs)
- `feature_stats_train.json` (per-feature mean, standard deviation, minimum, and maximum statistics, computed on the training split only).

All experiments are run using a fixed random seed of 7, shared across NumPy, PyTorch and the Optuna parameter sampler. `torch.backends.cudnn.deterministic` is set to `True` and `torch.backends.cudnn.benchmark` to `False` to disable the non-deterministic choice of convolution implementations by cuDNN. We will see results from these experiments that are identical to the reported metrics, if we first rerun `wb01` and `wb03c1` on a clean repository and compare their artefact file digests; these must be identical if the input files were the same.

5. Model Development

Building on the inputs and invariants established in Chapter 4, this chapter details the modelling apparatus that converted those elements into a classification engine and an explanation generator. The scope is confined to methods. The learning formulation is defined, the universal message-passing scheme is explicated, the candidate architectures are specified as if they were about to be implemented, and the training, searching, and interpretability procedures are described. No numerical findings appear here; all quantitative outputs will be presented in a single location in Chapter 6.

5.1 Problem formulation

Following Bellei et al. (2024), the Elliptic2 task is posed as binary subgraph classification. Let $G = (V, E)$ be a graph and $S^G \subseteq G$ a connected subgraph. Let $\mathcal{Y} = \{\text{licit}, \text{suspicious}\}$ be the label set and let $\mathcal{I}_{\text{train}}$ and $\mathcal{I}_{\text{test}}$ index disjoint collections of labelled subgraphs. Given the training data

$$\{(S_i^G, y_i) \mid S_i^G \subset G,\ y_i \in \mathcal{Y},\ i \in \mathcal{I}_{\text{train}}\}, \tag{5.1}$$

predict the label of each subgraph S_j^G for $j \in \mathcal{I}_{\text{test}}$. Throughout this thesis, a subgraph refers exclusively to a labelled connected component (`ccId`) and the encoding is 0 for licit and 1 for suspicious.

The evaluation scheme is established below and left unchanged for all experiments presented in the thesis. Area under the precision-recall curve (PR-AUC) is the main metric. This choice is dictated by the 2.27% positive rate: the area under the ROC curve is easy to make very high while the model detects almost no suspicious components. The remaining metrics are ROC-AUC, the F1 score at the threshold chosen from the validation set, and the entire confusion matrix. Thresholds are chosen on validation data by searching over a fine grid and picking the F1-maximiser, which is then fixed for evaluation on the test set, guarding against overfitting due to a threshold choice.

5.2 Message-passing framework

Every graph neural network considered in this thesis uses the message-passing paradigm of Gilmer et al. (2017). One layer updates every node v by

$$h_v^{(\ell+1)} = \text{UPDATE}^{(\ell)}\Big(h_v^{(\ell)}, \bigoplus_{u \in \mathcal{N}(v)} \text{MSG}^{(\ell)}\big(h_v^{(\ell)}, h_u^{(\ell)}, e_{uv}\big)\Big), \tag{5.2}$$

where MSG produces a message from every neighbour u (possibly conditioned on edge attribute e_{uv}), $\bigoplus$ aggregates the messages with a permutation-invariant function like mean, max, or sum, and UPDATE fuses those aggregate messages with the node's current state. Running L such layers allows any node to incorporate information from at most L hops away.

A pooling function POOL follows the last message-passing layer and compresses the node-level representations $\{h_v^{(L)} : v \in S^G\}$ to a single subgraph-level embedding that is fed to a multilayer perceptron MLP to produce a classification logit

$$\hat{y} = \mathrm{MLP}\Big(\mathrm{POOL}\big(\{h_v^{(L)} : v \in S^G\}\big)\Big). \tag{5.3}$$

With this outer structure held fixed, we can tune each building block in turn in the architecture experiment so that any change in performance can be confidently attributed to the block being tested rather than to unintended differences in the head, optimiser, or evaluation pipeline.

5.3 Architectures tested

Four message-passing layers were used as potential network bodies. GCN, GraphSAGE, and GATv2 were compared methodically in the primary architecture experiment. NNConv was probed as an edge-aware augmentation in a separate experiment, treated in detail in Section 5.7.

Graph Convolutional Network (GCN)

The GCN layer from Kipf and Welling (2017) applies a symmetric-normalised convolution over the neighbourhood, self-loops included:

$$h_v^{(\ell+1)} = \sigma\Big(\sum_{u \in \mathcal{N}(v)\cup\{v\}} \frac{1}{\sqrt{\hat{d}_v \hat{d}_u}} W^{(\ell)} h_u^{(\ell)}\Big), \tag{5.4}$$

where $\hat{d}_v = |\mathcal{N}(v)| + 1$ is the degree including the self-loop, $W^{(\ell)} \in \mathbb{R}^{d_{\text{out}} \times d_{\text{in}}}$ is a trainable matrix, and σ denotes a non-linearity. The symmetric normalisation renders GCN insensitive to differences in scale across nodes, but it ties together a node's own state and those of its neighbours via a single shared weight matrix, limiting expressivity on heterophilic graphs.

GraphSAGE

As defined by Hamilton et al. (2017), GraphSAGE employs two separate weight matrices to differentiate between a node's own embedding and the aggregated embedding of its neighbours:

$$h_v^{(\ell+1)} = \sigma\Big(W_{\text{self}}^{(\ell)} h_v^{(\ell)} + W_{\text{neigh}}^{(\ell)} \mathrm{AGG}\big(\{h_u^{(\ell)} : u \in \mathcal{N}(v)\}\big)\Big) \tag{5.5}$$

where AGG can represent mean, max, or LSTM aggregation functions. This paper utilises mean aggregation exclusively, implemented via the `SAGEConv` module within PyTorch Geometric. The distinction between the central node and its neighbours in this formulation suits the Elliptic2 dataset well, as fraudulent entities often exhibit a hub-and-spoke structure with a central node surrounded by leaf nodes. In such a configuration, preserving the central node's unique features distinct from the aggregated neighbour context helps retain critical information that a GCN would inadvertently merge.

GATv2 (Graph Attention Network v2)

GATv2 (Brody et al., 2022) substitutes standard aggregation with learned attention values, where the importance of a node u in v's neighbourhood depends dynamically on the features of both nodes u and v.

$$\alpha_{vu}^{(\ell)} = \frac{\exp\big(\vec{a}^{\top}\mathrm{LeakyReLU}\big(W^{(\ell)}[h_v^{(\ell)} \,\|\, h_u^{(\ell)}]\big)\big)}{\sum_{w\in\mathcal{N}(v)}\exp\big(\vec{a}^{\top}\mathrm{LeakyReLU}\big(W^{(\ell)}[h_v^{(\ell)} \,\|\, h_w^{(\ell)}]\big)\big)}, \tag{5.6}$$

$$h_v^{(\ell+1)} = \sigma\Big(\frac{1}{K}\sum_{k=1}^{K}\sum_{u\in\mathcal{N}(v)}\alpha_{vu}^{k,(\ell)}W_k^{(\ell)}h_u^{(\ell)}\Big), \tag{5.7}$$

where K denotes the number of attention heads, $\|$ indicates concatenation, and $\vec{a}$ is a learnable attention vector. This mechanism rectifies the static aggregation inherent in the original GAT (Veličković et al., 2018) by performing concatenation prior to the non-linearity rather than after, restoring the capacity to produce dynamic attention outputs. In the context of this thesis, GATv2 serves dual functions: the attention values are integrated into the training loss and are subsequently examined in Chapter 6 as a direct means of model interpretation.

NNConv

In Gilmer et al. (2017), the authors propose NNConv (Neural Message Passing with Convolution), where message passing utilises a small feed-forward neural network to determine a transformation matrix from an edge weight e_{uv}:

$$h_v^{(\ell+1)} = W^{(\ell)}h_v^{(\ell)} + \sum_{u\in\mathcal{N}(v)}F_\theta^{(\ell)}(e_{uv})\cdot h_u^{(\ell)} \tag{5.8}$$

where $F_\theta^{(\ell)} : \mathbb{R}^{d_e} \to \mathbb{R}^{d\times d}$ is a multi-layer perceptron. In comparison to the prior two architectures, NNConv is the most expressive layer in terms of the edge attributes it can encode; however, it comes with a significantly higher computational overhead. The number of trainable weights required for the feed-forward network F grows at $\mathcal{O}(d_e \cdot d^2)$ per layer, given that d_e corresponds to the number of features describing the edge attributes and d is the hidden dimension. It was these scaling constraints that informed the architectural choices presented in Section 5.7.

5.4 Pooling and refinement strategies

A pooling operation aggregates a variable number of node embeddings into a single fixed-size representation of a subgraph. The subsequent architecture comparisons are conducted using one of two approaches. Global max-pooling takes the maximum element value in each node-embedding dimension. This operation is non-learnable, permutation invariant, and tends to emphasise the single most salient node for a given feature dimension. Attention pooling (Li et al., 2016) learns a gating network $g_\phi : \mathbb{R}^d \to \mathbb{R}$ to generate a single, scalar importance score for each node.

$$\mathrm{POOL}_{\mathrm{att}}\big(\{h_v\}_{v\in S^G}\big) = \sum_{v\in S^G}\mathrm{softmax}_v\big(g_\phi(h_v)\big)\cdot h_v \tag{5.9}$$

The benefit of this approach is the ability to learn a node importance distribution specific to each subgraph, at the expense of a few extra learnable parameters. Like the attention weights from Graph Attention Networks, the gate weights are examined as an interpretable dimension in Chapter 6.

The Jumping Knowledge (JK) network proposed by Xu et al. (2018) is intended as a refinement to the limited effective receptive field resulting from a fixed number of message-passing layers. After L steps in an L-layer model, each node's final representation is obtained as the concatenation (or element-wise maximum) over its representations from each layer $1 \leq \ell \leq L$:

$$h_v^{\mathrm{JK}} = \text{JK-Agg}\big(h_v^{(1)}, h_v^{(2)}, \ldots, h_v^{(L)}\big). \tag{5.10}$$

The JK-cat option concatenates each node's representations from the various layers and leaves the reweighting to the classification head. JK-max is parameter-free and uses element-wise maximum. Neither option is used as part of our pre-processing pipeline in Chapter 4; both options are introduced as a potential refinement during the model architectural search phase (Section 5.7).

5.5 Training protocol

All models were trained using a binary cross-entropy loss over the 2 logits with class weighting based on inverse class frequency, per Section 4.5. The class weights were calculated using only the training data and then the same weights were applied to the validation data as well as the test data. The models were optimised using Adam (Kingma and Ba, 2015) with a base learning rate determined by hyperparameter search and weight decay was set to 10^{-4}. For both training and evaluation, the batch size was set to 256 subgraphs and 512 subgraphs respectively.

We monitored validation PR-AUC for early stopping (patience of 15 epochs, maximum budget of 80 epochs) and restored the best checkpoint (based on validation PR-AUC, rather than loss) at the end of training to test on the held-out test data. It is important to restore a model based on validation PR-AUC rather than validation loss due to class imbalance, since the validation loss is dominated by the majority class and could still be decreasing long after the performance of the model (as measured on the validation test set) has stalled.

The runs were all deterministic. We used fixed random seeds for numpy, PyTorch (both CPU and CUDA) and the Optuna sampler at the beginning of each run. For CUDA operations we set `torch.backends.cudnn.deterministic` to `True` and `torch.backends.cudnn.benchmark` to `False` to avoid non-deterministic kernel auto-tuning. Finally, we used `CUBLAS_WORKSPACE_CONFIG=:4096:8` to disable any non-determinism in cuBLAS operations. Running the same code on the same hardware and software environment and code version produces bit-identical results.

5.6 Hyperparameter search

We used the Optuna (Akiba et al., 2019) hyperparameter optimiser with the TPE sampler of Bergstra et al. (2011) and a median pruner for our hyperparameter searches. We

conducted a separate search for each architecture with a fixed seed for the sampler. We set the median pruner with `n_startup_trials=5` and `n_warmup_steps=10`, where trials are pruned if their intermediate validation PR-AUC at the epoch ten is below the median of all non-pruned trials at that epoch.

For the main architecture study we used the following hyperparameter search space:

- `hidden_dim`: $\{64, 128, 256\}$
- `num_layers`: $\{2, 3\}$
- `dropout`: $[0.0, 0.5]$ at steps of 0.05
- `lr`: $[10^{-4}, 5 \times 10^{-3}]$ on a log-scale
- `pool`: $\{\text{mean}, \text{max}\}$
- `heads`: $\{1, 2, 4\}$ (GATv2 only)

The hyperparameter search space for the architectural refinement ablation also included:

- `jk_mode`: $\{\text{none}, \text{cat}, \text{max}\}$
- `pool`: $\{\text{max}, \text{attention}\}$
- `lr_scheduler`: cosine on/off

The number of hyperparameter trials was set to 30 for the main architecture study and 10 for the refinement ablation, given a smaller search space. We use fANOVA (Hutter et al., 2014) to calculate feature importances on the completed trials, to see which hyperparameters contribute the most to the validation PR-AUC (rather than finding the optimal hyperparameter values).

Reported importances and sensitivities are stated in Chapter 6.

5.7 Experimental phases and rationale

Model construction was broken into four discrete phases, each phase arising in response to a question that the previous phase had allowed to be asked in a more well-formed way.

1. **Phase Wb02: baselines.** Two baseline models were built prior to the architecture search. The first was a logistic regression on pooled subgraph features (i.e. mean, max and standard deviation over the 43 node features) which indicated the signal available via tabular aggregation of graph data, with no relational reasoning. The second was a default two-layer GraphSAGE model with a hidden dimension of 64 and global mean pooling, indicating the value of a GNN over and above a tabular model. These baselines employed the same class-weighting, split and thresholding configuration as all later experiments, such that their results are directly comparable.

2. **Phase Wb03: systematic architecture comparison.** The main architecture investigation compared GraphSAGE, GCN and GATv2, under a common search space. Thirty trials per architecture were launched by Optuna with a median pruner, for 90 trials in total; roughly ten trials per architecture (about 30 surviving runs in total) completed without being pruned. The main model was chosen by comparing test PR-AUC among the best surviving model per architecture and its corresponding checkpoint was saved for explainability work in Chapter 6.

3. **Phase Wb03b: architectural refinement.** The second study further restricted the architecture pool to GATv2 and GraphSAGE (GCN having been discarded on the basis of Wb03 results), and added JumpingKnowledge, attention pooling and the cosine learning rate scheduler to the search space. The refinement study also included a controlled ablation across four corners of the refinement space, isolating the effect of each refinement variable by fixing the remainder at the configuration associated with the Wb03 winner model. This refinement was run a second time, with strict checkpointing, and only this second run was kept for the explainability work.

4. **Phase Wb03c: edge-feature integration.** Phase Wb03c was a pre-registered study conducted on the 95 edge features registered in Section 4.3 in three stages. Stage Wb03c1 involved the same preprocessing as in Chapter 4, which is documented in Chapter 4. Stage Wb03c2 involved the search for an optimal neural network architecture for NNConv, which used the following early-stopping criterion: stop early once it became clear no NNConv trial was approaching the validation PR-AUC of 0.50 reached by the GATv2 family. The early-stopping criterion was determined based on the NNConv parameterisation discussion in Section 5.3, where the number of parameters in NNConv is $\mathcal{O}(d_e \cdot d^2)$, where d is the degree of a subgraph node, and d_e is the number of edges. Based on the median degree of 3 for Elliptic2, this means that NNConv has too many parameters for an acceptable ratio between number of parameters and informative neighbours' count. Five models were completed before the stopping criterion was applied, which yielded a PR-AUC of 0.4063 on the validation set, at which point the early stopping criterion was triggered, and thus we proceeded no further on this stage. In Wb03c3, two architectures that used the edge-aware message passing scheme, GINEConv (Hu et al., 2020) and TransformerConv (Shi et al., 2021), were run with 30 and 23 trials respectively. GINE hit the 0.50 validation PR-AUC threshold and was continued until the search was exhausted, with the final PR-AUC being 0.5393 and 0.5087 on the validation and test set respectively, with the former being within noise of the GATv2 + attention pool model in Wb03b, which was 0.5421 and 0.515 on the validation and test set respectively. TransformerConv did not pass the 0.50 validation PR-AUC threshold, which was achieved at 0.4307. Full search space tables and best performing configurations per model for all three sub-phases are reported in Appendix A.

The Wb03c3 outcome proves the initial model selection. GINEConv's final test performance yielded a PR-AUC of 0.5087, which falls short of the 0.5148 PR-AUC achieved by the GATv2 attention-pool model, while TransformerConv fails to even meet the threshold value of 0.5 (highest validation score of 0.4307).

When constructing future models, if the approach emphasises edge-feature incorporation as opposed to the GATv2 attention-readout scheme, the most suitable starting point

would be the GINEConv checkpoint.

Treating such a stopping rule as a study outcome and registering it beforehand is done to avoid post-hoc rationalisation, which is a well-known pitfall in empirical ML. A registered rule requires the author to commit to a decision without knowing the result.

5.8 Explainability methodology

The primary model from Phase Wb03b forms the focus of the explainability study. This framework does not attempt to provide one definitive account for each prediction; rather, it relies on triangulating multiple techniques that fail in distinct ways, with the expectation that no post-hoc method is fully credible on its own (Agarwal et al., 2022). The process unfolds in three tiers.

At Tier 0, the sample selection level, explanations were calculated on a stratified subset of 200 subgraphs extracted from the test set. This stratification accounts for two factors: the four prediction categories, true positives (themselves split into high-confidence and borderline based on predicted probability), true negatives (divided into near-threshold and random), false positives, and false negatives, and the subgraph size ranges (2 nodes, 3 nodes, and 4 or more nodes). A stratified approach is necessary due to Elliptic2's strongly right-skewed subgraph size distribution; otherwise, a purely random sample would capture few large components and almost no borderline predictions.

Tier 1 addresses feature attribution. For each subgraph in the selected sample, we applied two techniques: Integrated Gradients (IG) (Sundararajan et al., 2017), implemented with Captum's IG class and a baseline of all-zero tensors (50 integration steps); and KernelSHAP (Lundberg and Lee, 2017), implemented with KernelExplainer using 512 training-set examples as the background. Cross-method consistency is quantified by computing the per-feature Spearman correlation between the rankings generated by IG and SHAP, both in aggregate and on a case-by-case basis. We include these metrics because a highly precise feature attribution that cannot be replicated by another method may be a reflection of the method itself, not the model.

At Tier 2, the structural explanation level, we employed two post-hoc methods. GNNExplainer (Ying et al., 2019) was trained for each prediction using default sparsity and size regularisation parameters; its output is a soft weight assigned to each edge in the subgraph. SubgraphX (Yuan et al., 2021) was applied to a limited set of 50 samples, 10 from each prediction category, given the expense of its Monte Carlo tree-search algorithm; its output is a discrete set of nodes whose induced subgraph maximises the model's predicted class probability. In addition to these post-hoc methods, we extracted the attention weights directly from the GATv2 model by collecting the layer-wise attention coefficients and pooling-gate scores. Attention does not qualify as an explanation method in the Yuan et al. sense, but rather as another model readout channel, which we report in terms of Jaccard coefficient with the post-hoc explanation for each sample in Chapter 6 as an additional diagnostic.

Fidelity and sparsity of the subgraph explanations are computed using two metrics, Fidelity+ and Fidelity-, each expressed here as a conventional form of the average (Yuan

et al., 2023):

$$\text{Fidelity+} = \frac{1}{N}\sum_{i=1}^{N}\big(f(G_i) - f(G_i \setminus S_i)\big), \qquad \text{Fidelity-} = \frac{1}{N}\sum_{i=1}^{N}\big(f(G_i) - f(S_i)\big). \quad (5.11)$$

Here, $f(G)$ is the model's predicted probability for the target class in the original graph. $G_i \setminus S_i$ is the graph resulting from masking out S_i from graph G_i, and S_i is the subgraph that constitutes the explanation for graph G_i. Fidelity+ quantifies the drop in the model's predicted probability when the explanation is removed; Fidelity- quantifies how well the explanation itself preserves the model's prediction. Sparsity is the proportion of edges retained in the GNNExplainer explanation, or the proportion of nodes retained in the SubgraphX explanation.

Chapter 5 enumerates every design choice that differentiates the analyses presented in this study. These decisions encompass the problem specification, the model architectures, pooling and feature refinement strategies, the model training procedure, the design of the hyperparameter search, the three main experimental stages (including their pre-registered decision rules), and the explainability procedure. Chapter 6 details the results of these experiments.

6. Performance and Model Outcome

6.1 Introduction

In this chapter, we present the empirical results of the system described in Chapter 5 across three distinct performance dimensions: its predictive performance as gauged on the Elliptic2 benchmark; its feature-level interpretability; and its structural level interpretability. It is important to note that the assertions made within this chapter are relatively limited in scope. On the predictive performance side, we evaluate the model on the test split held out of Elliptic2, which was constructed using the 70/10/20 stratified component-level split described in Section 4. In terms of the interpretation performance, we evaluate on a stratified sample of 200 test subgraphs that encompasses the entire confusion matrix, to determine whether explanations are homogeneous over true positives, false positives, false negatives, and random true negatives. If it is not otherwise mentioned, the specific model under study is the primary model that was selected in Section 6.2: a two-layer GATv2 with a 256-dimensional hidden representation, one attention head, a dropout rate of 0.2, and a global attention pooling head, trained using binary cross-entropy loss at a learning rate of approximately 7.5×10^{-4}, for a total of 288,515 trainable parameters.

6.2 Model evaluation

6.2.1 Architecture sweep

Before discussing the architecture comparison itself, the predictive value of using a graph neural network at all is established by the Wb02 baselines. A pooled logistic regression on the mean, max and standard deviation of the 43 node features achieves a test PR-AUC of 0.154; the default two-layer GraphSAGE model with global mean pooling reaches 0.401 on the same split. The 0.247-point absolute gain (a relative improvement of approximately 161%) shows that a substantial portion of the Elliptic2 signal lives in the local subgraph topology rather than in the per-node feature marginals, and that the structural information is recoverable through message-passing in a way that pooled tabular features cannot reproduce. The same transition reduces false positives from 889 to 417 while increasing true positives from 207 to 253, which is an operational improvement and not just a metric one.

Three graph neural network (GNN) architectures, namely GraphSAGE, GCN, and GATv2, were evaluated using the same pre-processing steps, the same sampling approach, and the same computational budget for the hyperparameter search algorithm. Each GNN architecture was subjected to a 30-run Bayesian optimisation search via Optuna with a median pruner to optimise the hidden dimension, number of layers, dropout rate, learning rate, the type of pooling, and the number of attention heads as required. Among the three resulting models, we use the test PR-AUC metric as the model selection criterion; we use this specific metric instead of, say, test AUC-ROC because PR-AUC is generally more relevant to our use case given that only 2.27% of the instances are from the positive class (or are suspicious subgraphs), and thus, AUC-ROC may overestimate the actual

performance.

The three best performing models were extremely close to each other, with GATv2 being the highest ranked, achieving a PR-AUC score of 0.496 on test data, followed by GraphSAGE and GCN, achieving PR-AUC scores of 0.485 and 0.420, respectively. Each of the top three ranked models was within a difference of approximately eight points from one another on PR-AUC and within one point on the ROC-AUC. As such, the fact that all three models are close is interesting in its own right, and indicates that once node features are provided, the choice of GNN architecture on this task is a secondary consideration. The more critical component that drives the models' success is the access to the node features and the local aggregation paradigm of the subgraphs, which we hold constant throughout all three models.

Figure 6.1 shows the precision-recall and ROC behaviour of the three architectures on the test split, with the curves overlapping closely along most of the operating range and GCN trailing only in the mid-recall region. Figure 6.2 reports the corresponding confusion matrices at the per-architecture F1-optimal thresholds. The numbers in this confusion matrix figure correspond to the architecture-sweep best models prior to the attention-pool refinement of Section 6.2.2; the canonical figures used elsewhere in this thesis derive from the refined GATv2 + attention pool model, whose retraining is shown in Figure 6.5.

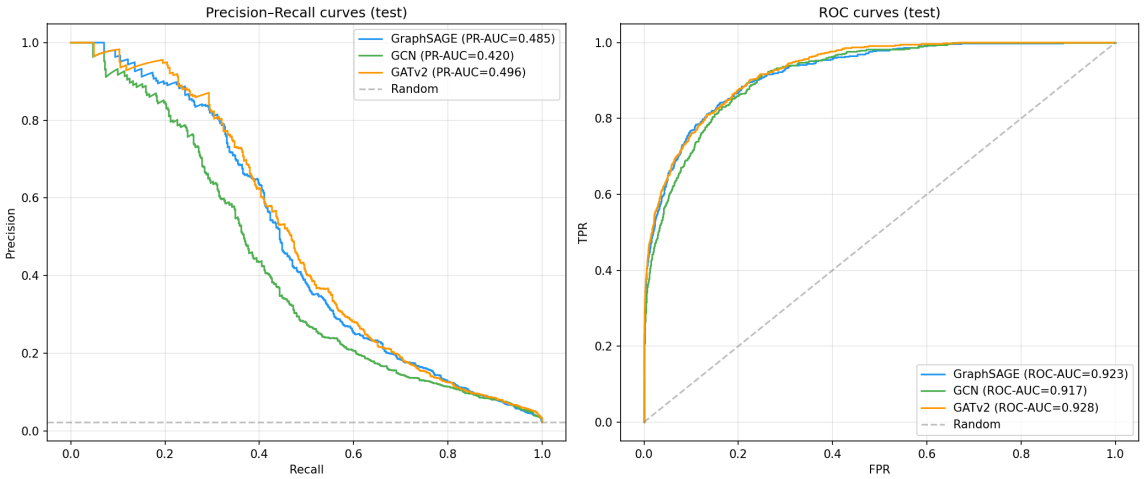

Figure 6.1: Precision-recall and ROC curves on the Elliptic2 test split for the best GraphSAGE, GCN, and GATv2 model from the Wb03 architecture sweep, prior to attention-pool refinement. Random baseline shown as a dashed line.

As GATv2 is the top performer based on the PR-AUC scores, in the following runs we replaced the standard pooling head of GATv2 with the gated attention pooling operator of Li et al. (2016). The gated attention pooling operator raised the PR-AUC score on the test data to 0.515 and the F1 score to 0.516 compared to the initial GATv2 model. This particular version of the gated attention pooling will be used in all the subsequent tests that do not require changing of the GNN layers. Gated attention pooling is a preferred option for this particular use case because the subgraph samples have a node size range of 2–296 (median of 3), and a trainable node-level importance weight is able to handle such varying node counts better than any type of global pooling such as mean-pooling or max-pooling, which are insensitive to the node's identity. This specific version of the gated attention pooled GATv2 model will be used as our primary model in this chapter unless it is noted otherwise.

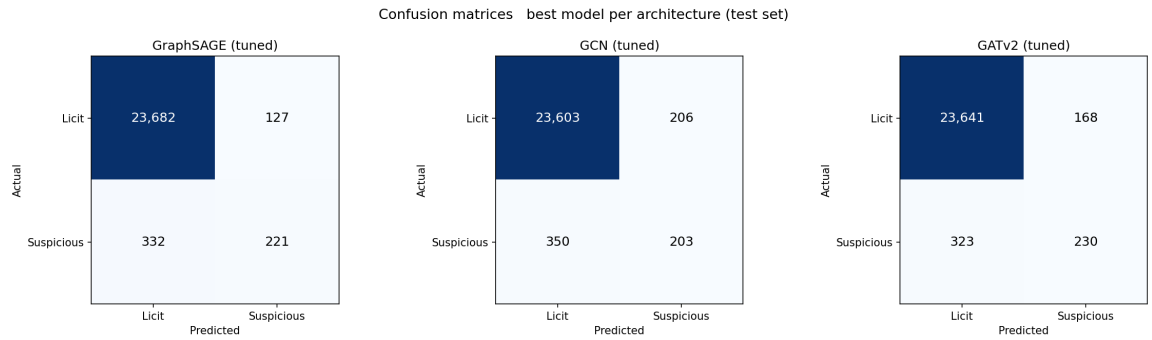

Figure 6.2: Confusion matrices for the best Wb03 model per architecture (GraphSAGE, GCN, GATv2 with their default pooling head) on the test split, evaluated at the per-architecture F1-optimal threshold. These matrices precede the attention-pool refinement; the canonical GATv2 + attention pool numbers (TP = 237, FP = 128, FN = 316, TN = 23,681) are reported in Section 6.2.3.

The Wb03b refinement search, restricted to GATv2 with attention pooling and an extended search space (JumpingKnowledge mode, learning-rate scheduler, larger hidden dimensions), was conducted using a TPE sampler with 35 trials. Figure 6.4 shows the resulting optimisation history and the fANOVA parameter importance, where the learning rate dominates the importance ranking, followed by the pooling choice and the Jumping-Knowledge mode. Architecture identity itself contributes negligibly, which is consistent with the architecture sweep showing the three GNN backbones within approximately eight points of one another. The best trial in the refinement search reached a validation PR-AUC of 0.5423.

The best Wb03b configuration was retrained from scratch with strict checkpointing, and only this second run was used as the primary model for the explainability work in subsequent sections. Figure 6.5 reports the training-loss, validation PR-AUC, and validation ROC-AUC trajectories across the retraining run. The validation PR-AUC peaks at 0.5421 on epoch 63, marginally above the original Wb03b best of 0.5344, with the validation ROC-AUC plateauing in the 0.92–0.94 range from approximately epoch 20 onwards. The retrained checkpoint at epoch 63 is the model evaluated on the test split (PR-AUC 0.515) and is the same checkpoint against which all explainability methods in Section 6.3 and 6.4 are run.

The cumulative test-set picture across all three experimental phases is summarised in Table 6.1. Two within-experiment patterns are notable. First, all three Wb03 tuned architectures comfortably outperform the default Wb02 GraphSAGE baseline, indicating that a meaningful share of performance comes from optimisation rather than architecture alone. Second, the Wb03b ablation locates the best test-set PR-AUC at the GATv2 + attention pool configuration, which is the model used as the primary checkpoint in the explainability work of Sections 6.3 and 6.4. The JK(cat) variant trades approximately four points of PR-AUC for an additional 4.6 percentage points of recall, and is discussed as the recall-prioritised alternative in Chapter 9.

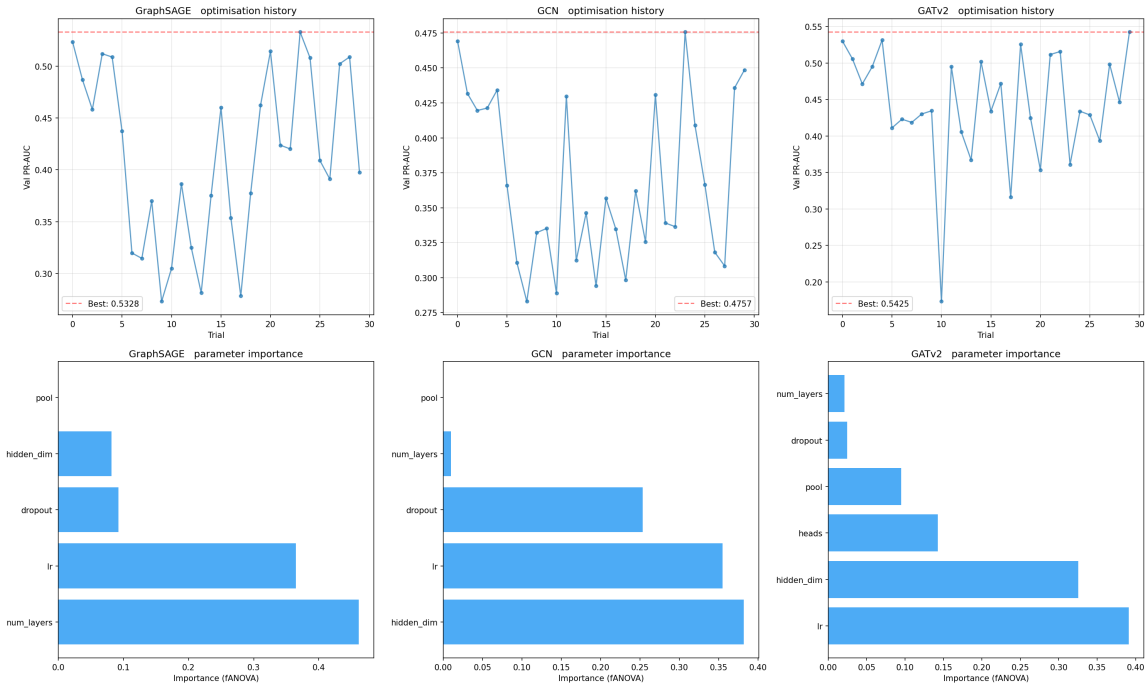

Figure 6.3: Wb03 Optuna search diagnostics for the three architectures. Top row: validation PR-AUC trajectory across 30 trials, with the best trial marked. Bottom row: fANOVA parameter importance ranking. Learning rate dominates importance for GATv2 and GraphSAGE; GCN distributes importance more evenly across hidden dimension, learning rate, and dropout.

Table 6.1: Cumulative test-set comparison across experimental phases. The GATv2 + attention pool row reports the canonical retrain numbers used throughout this thesis; the other Wb03b ablation rows are reported from the original ablation pass. Bold marks the per-column best in the Wb03b group.

Group	Model	PR-AUC	ROC-AUC	F1	Precision	Recall	Params
Wb02	LogReg (pooled)	0.154	0.890	0.251	0.189	0.374	—
Wb02	GraphSAGE (default)	0.401	0.914	0.414	0.378	0.458	—
Wb03	GraphSAGE (tuned)	0.485	0.923	0.491	0.635	0.400	94,466
Wb03	GCN (tuned)	0.420	0.917	0.422	0.496	0.367	144,386
Wb03	GATv2 (tuned)	0.496	0.928	0.484	0.578	0.416	222,466
Wb03b	GATv2 + max pool	0.489	0.928	0.492	0.639	0.400	222k
Wb03b	**GATv2 + attention pool**	**0.515**	**0.934**	0.516	**0.649**	0.429	288k
Wb03b	GATv2 + JK(cat)	0.516	0.932	0.504	0.575	**0.449**	288k
Wb03b	GATv2 + attention + JK(cat)	0.496	0.932	**0.506**	0.629	0.423	551k

Generalisation must be discussed alongside these headline numbers, because the study uses a single fixed hold-out test split rather than repeated cross-validation. Three controls limit the resulting risk of overfitting to the model-selection process. First, all model selection decisions are made on the validation split, with the F1-optimal threshold transferred unchanged to the test set. Second, the same data partition is reused across every experiment in this thesis, so architecture comparisons are not contaminated by split-to-split noise. Third, the validation-to-test gap itself functions as a descriptive overfitting signal. Table 6.2 reports this gap for each Wb03b variant. The GATv2 + attention pool model

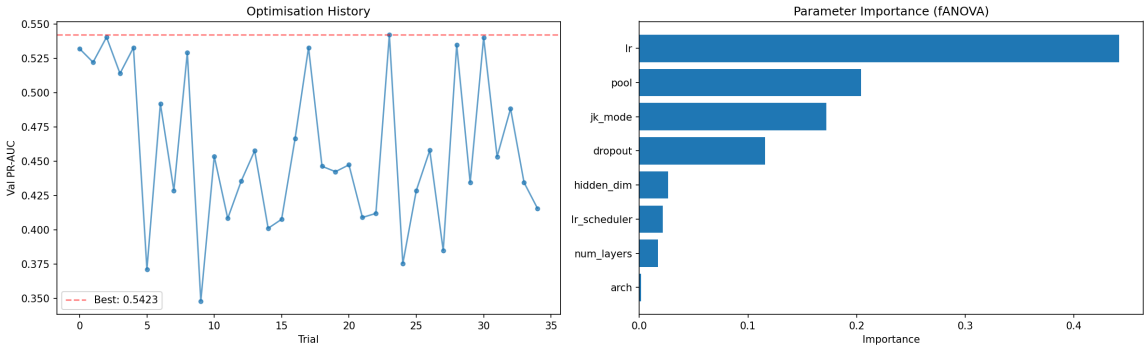

Figure 6.4: Wb03b refinement search diagnostics, restricted to GATv2 with attention pooling and an extended hyperparameter space. Left: validation PR-AUC across 35 trials. Right: fANOVA parameter importance, dominated by the learning rate.

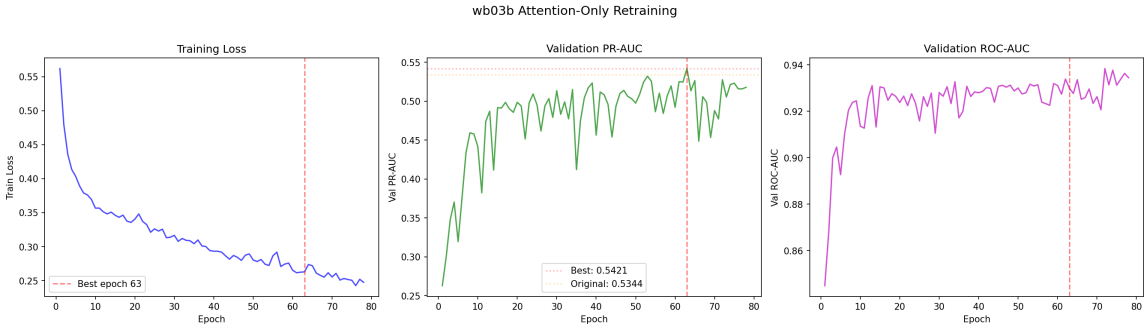

Figure 6.5: Training trajectory of the Wb03b GATv2 + attention pool retrain selected as the primary model. Left: training loss across epochs. Middle: validation PR-AUC, with the best epoch (63, val PR-AUC 0.5421) marked and compared against the original Wb03b best (val PR-AUC 0.5344). Right: validation ROC-AUC. The epoch 63 checkpoint is the model used in the test-set evaluation (test PR-AUC 0.515) and in all explainability experiments.

(canonical retrain) shows a gap of 0.027, which is wider than the gap of 0.009 observed for GATv2 + JK(cat), but tighter than the 0.054 gap of the max-pool baseline and the 0.044 gap of the four-way attention + JK combination. The narrow gap on the simpler ablation variants and the substantially larger gap on the 551k-parameter combined model are consistent with mild hyperparameter-search overfitting in the larger search space rather than with catastrophic memorisation.

Table 6.2: Validation-to-test PR-AUC gap on the Wb03b ablation variants. Smaller gaps indicate more stable transfer from model selection to hold-out evaluation. The attention pool row reports the canonical retrain (validation PR-AUC 0.5421, test PR-AUC 0.515); the remaining rows report the original Wb03b ablation pass.

Variant	Val PR-AUC	Test PR-AUC	Gap	Precision	Recall	Params
Max pool (baseline)	0.5429	0.4887	0.0542	0.6387	0.3996	222k
Attention pool (canonical retrain)	**0.5421**	**0.515**	0.0273	**0.6493**	0.4286	288k
JK cat only	0.5245	0.5157	**0.0088**	0.5754	**0.4485**	288k
Attention + JK	0.5398	0.4963	0.0435	0.6290	0.4231	551k

6.2.2 Comparison to published baselines

Table 6.3 compares the best tuned subgraph-feature models in this work against the three models reported in Bellei et al. (2024). Before considering the actual numerical values, note the following two caveats. First, the paper's baselines (GNN-Seg, Sub2Vec, GLASS) were trained without node or edge features, because the authors ran their experiments on CPU-only hardware unable to hold the full feature tensor in memory. Second, GLASS additionally exploits the 196M-edge background graph through its 0–1 labelling trick, while GNN-Seg and Sub2Vec do not. The comparison therefore contrasts a node-feature subgraph-local regime against a structure-only regime, with GLASS the strongest available structure-only baseline.

Table 6.3: Test-set performance on Elliptic2. "Features" indicates whether the 43-dimensional node features were used. "Background graph" indicates whether the 49M-node background graph was exploited during training.

Model	Source	Features	Background graph	PR-AUC	ROC-AUC
GNN-Seg	Bellei et al., 2024	No	No	0.026	0.537
Sub2Vec	Bellei et al., 2024	No	No	0.022	0.496
GLASS	Bellei et al., 2024	No	Yes	0.208	0.889
GraphSAGE (tuned)	This work	Yes	No	0.485	0.923
GCN (tuned)	This work	Yes	No	0.420	0.917
GATv2 (tuned)	This work	Yes	No	0.496	0.928
GATv2 + attention pool	**This work**	**Yes**	**No**	**0.515**	**0.934**

The primary model reaches a test PR-AUC of 0.515 against GLASS's 0.208, a 2.47× increase on the headline metric under class imbalance. The three tuned backbones in this work all exceed GLASS by at least a factor of two on PR-AUC: GraphSAGE by 2.33×, GCN by 2.02×, GATv2 by 2.39×, and the refined GATv2 by 2.47×. Read as a family, these numbers support a specific claim: the discriminative signal in Elliptic2 is concentrated in node attributes rather than in macro-topology. A feature-aware subgraph-local classifier trained in under half an hour on a single GPU outperforms a structure-only method trained across many hours on a large CPU cluster. This is the central empirical result of the thesis.

6.2.3 Operating characteristics

The threshold for binarising predictions was chosen on the validation split to maximise F1, yielding 0.9128. At that threshold, the canonical GATv2 + attention pool retrain gives a confusion matrix of TP= 237, FP= 128, FN= 316, TN= 23,681, corresponding to precision 0.649, recall 0.429, and F1 0.516. The high threshold reflects the base-rate skew: to achieve a precision above 0.5 on a 2.27% positive class, the classifier must be confident before committing. A lower threshold would raise recall at the cost of drowning suspicious predictions in false positives, which in a compliance setting is arguably the worse of the two errors: an investigative team that receives a thousand alerts a week and finds 90% to be licit will stop acting on alerts.

The deployment cost of each model is more clearly seen by reading the confusion matrices alongside two operational quantities. The false-positive rate (FPR) is the fraction

of licit subgraphs that the model flags, and is the metric that governs alert fatigue once the licit population is large. The alert rate is the fraction of *all* test-set subgraphs that the model flags, and is the metric that governs analyst capacity planning. Table 6.4 reports both alongside the test-split confusion matrix for every model considered in this chapter.

Table 6.4: Operational burden on the test set at the F1-optimal decision threshold. FPR is computed against the 23,809 licit subgraphs; the alert rate is computed against the full 24,362-subgraph test split. The GATv2 + attention pool row reports the canonical retrain checkpoint used in the explainability experiments of Sections 6.3 and 6.4.

Model	**TN**	**FP**	**FN**	**TP**	**Threshold**	**FPR**	**Alert rate**
LogReg (pooled)	22,920	889	346	207	—	3.73%	4.50%
GraphSAGE (default)	23,392	417	300	253	—	1.75%	2.75%
GraphSAGE (tuned)	23,666	143	338	215	0.9597	0.61%	1.47%
GCN (tuned)	23,603	206	350	203	0.9265	0.87%	1.68%
GATv2 (tuned)	23,641	168	323	230	0.8775	0.71%	1.63%
GATv2 + max pool	23,684	125	332	221	0.9176	0.53%	1.42%
GATv2 + attention pool	**23,681**	128	316	**237**	0.9128	0.54%	1.50%

Two patterns deserve comment. First, every Wb03 tuned model reduces the alert rate by roughly two-thirds relative to the pooled logistic regression baseline (4.50% to between 1.42% and 1.68%), while simultaneously increasing the true-positive count or holding it close to the default GraphSAGE level. The combined effect is an alert queue that an analyst team could plausibly process: at a 1.50% alert rate, the canonical model produces 365 alerts on a 24,362-subgraph test split, of which 237 are genuine. Second, the canonical GATv2 + attention pool model is operationally close to the GATv2 + max pool baseline (FPR 0.54% versus 0.53%) but recovers 16 additional true positives at the same threshold and reduces the false-negative count by 16, which is the headline operational improvement attributable to the pooling change.

The choice of which Wb03b variant to deploy is itself a decision, and the relevant axis is the precision-recall trade-off. The canonical GATv2 + attention pool checkpoint sits at the precision-first end: at 0.649 precision, a flagged subgraph is approximately 29× more likely to be suspicious than a uniformly drawn one given the 2.27% base rate, which is the operating regime appropriate to a compliance team with bounded investigative capacity and a high cost per false alarm. The GATv2 + JK(cat) variant sits at the recall-first end: it reaches recall 0.449 (approximately 248 suspicious detections out of 553) at the cost of a lower precision (0.575) and a correspondingly larger alert queue. The combined attention + JK variant doubles the parameter count to 551k without delivering a commensurate improvement on either metric, so its extra capacity is not justified by the observed evidence. The choice between the two single-mechanism variants is therefore not a question of which is better in absolute terms but of which operating point the deployer is targeting; this point is revisited in the deployment discussion of Chapter 9.

Precision-recall behaviour across the operating range is shown in Figure 6.6. The curve sits well above the majority-class baseline of 0.0227 across all recalls, with a sharp drop in precision below recall 0.6. This shape is consistent with a model that has learned a compact set of high-confidence suspicious patterns but does not generalise to the full illicit population. The 316 false negatives suggest that a meaningful fraction of suspi-

cious subgraphs present node-feature profiles that overlap with licit activity under the 43-dimensional encoding. Closing that gap is likely a feature-engineering problem rather than an architectural one.

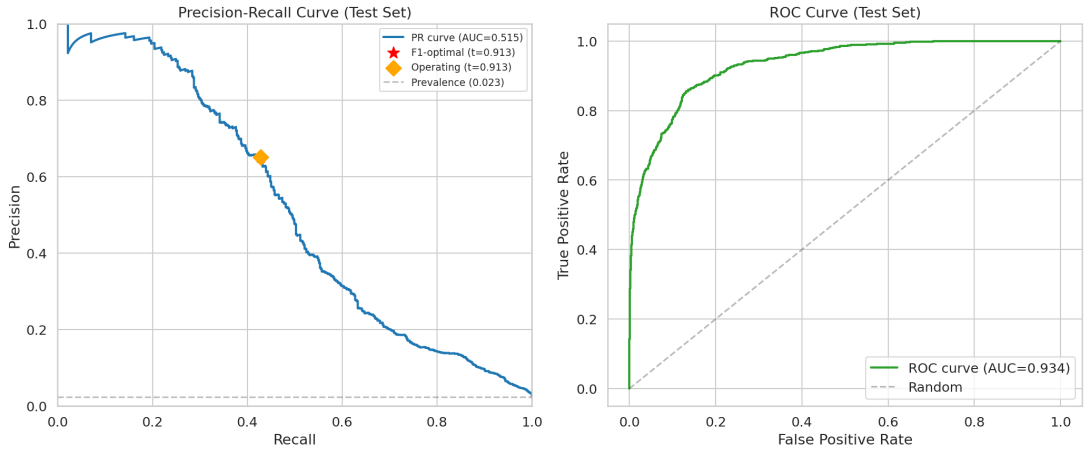

Figure 6.6: Precision-recall and ROC curves for the primary model (GATv2 + attention pool) on the Elliptic2 test split. The dashed line marks the majority-class baseline (0.0227).

6.2.4 What the numbers do and do not support

The model's predictive performance relative to published baselines is strong and consistent across three architectures. What this does not establish is that the model would generalise to a different blockchain, a different time window, or a different labelling procedure. The subgraphs utilised by Elliptic2 were derived via a deliberate traversal procedure (spanning a maximum of six hops from a detected malicious subgraph over a period of one year, as determined by the path-typology filter (Bellei et al., 2024)), and thus the learned signals of the model are strictly a function of that procedure. A 2.47× increase in the PR-AUC score for the method is only relevant to the task as defined by Elliptic2, and not a real-world AML application. This point regarding real-world deployment will be revisited in Chapter 8.

6.3 Explainability on the feature level

6.3.1 Protocol and method selection

All explainability experiments in Sections 6.3 and 6.4 are run on the same stratified 200-subgraph sample drawn from the test split. Figure 6.7 reports the composition of this sample across the six prediction quadrants (40 confident true positives, 40 false positives, 40 false negatives, 30 random true negatives, 30 near-threshold true negatives, 20 borderline true positives), the size buckets (2-node, 3-node, and 4+-node, evenly populated within ±5 subgraphs), and the predicted suspicious-class probability distribution per quadrant relative to the F1-optimal threshold of 0.913. The stratification ensures coverage of the entire confusion matrix and the full subgraph-size range, neither of which

a uniform random draw would have provided given the right-skewed size distribution and the 43:1 class imbalance.

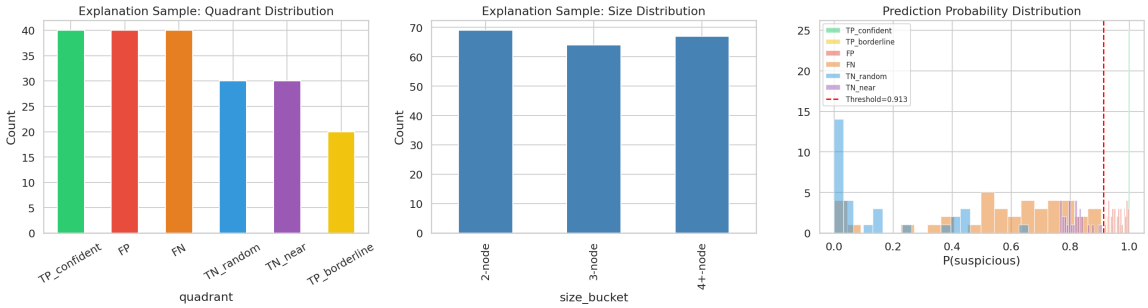

Figure 6.7: Composition of the 200-subgraph stratified sample used in all Section 6.3 and 6.4 explainability experiments. Left: per-quadrant counts. Middle: size-bucket counts. Right: distribution of the model's predicted suspicious-class probability per quadrant, with the F1-optimal threshold of 0.913 marked.

Two feature-attribution methods are evaluated, Integrated Gradients (Sundararajan et al., 2017) and Kernel SHAP (Lundberg and Lee, 2017). Their selection is based on complementarity: IG is a path-integral feature attribution scheme based on model gradients, and SHAP is a sampling-based approximation of Shapley values based on model outputs. With distinct assumptions, computations, and underlying frameworks, agreement between them is more informative than agreement between two attribution methods of the same type (e.g. two gradient-based methods). We focus the attribution on the scalar logit for the suspicious class as predicted for a given subgraph. The baseline for IG was zero and, for SHAP, we used a background of a randomly selected sample of subgraphs balanced for the two classes. Attributions are calculated for the same stratified sample of 200 subgraphs used throughout the explainability experiments. For each feature dimension, we subtracted the mean of the attribution of correctly classified suspicious subgraphs from the mean of the attribution of correctly classified licit subgraphs. We ordered the result by absolute value to obtain the "discriminative channels", features whose attributed value differs significantly between the two classes.

6.3.2 Discriminative channels

Figure 6.8 shows the top-20 features with the highest magnitude of the signed difference across methods.

The top of the ranking is shared by five features: F23, F27, F35, F19, F29. Two caveats are in order to avoid misinterpretation. First, although F35 and F19 are the most salient channels in terms of magnitude, we note that, in absolute terms, their attribution to licit subgraphs exceeds their attribution to suspicious subgraphs; thus, a reader that looked only at the attribution magnitudes would be led to mistakenly believe that F35 and F19 drive suspicious predictions. However, the contribution that a feature makes to the prediction comes from the relative, signed difference in attribution between the classes, not the attribution magnitude. We highlight F23 (negative difference: stronger on the licit class) and F27 (positive difference: stronger on the suspicious class), followed by F35 (negative difference), F19 (negative difference), and F29 (negative difference). The tendency to treat attribution as a measure of class membership rather than as a signed

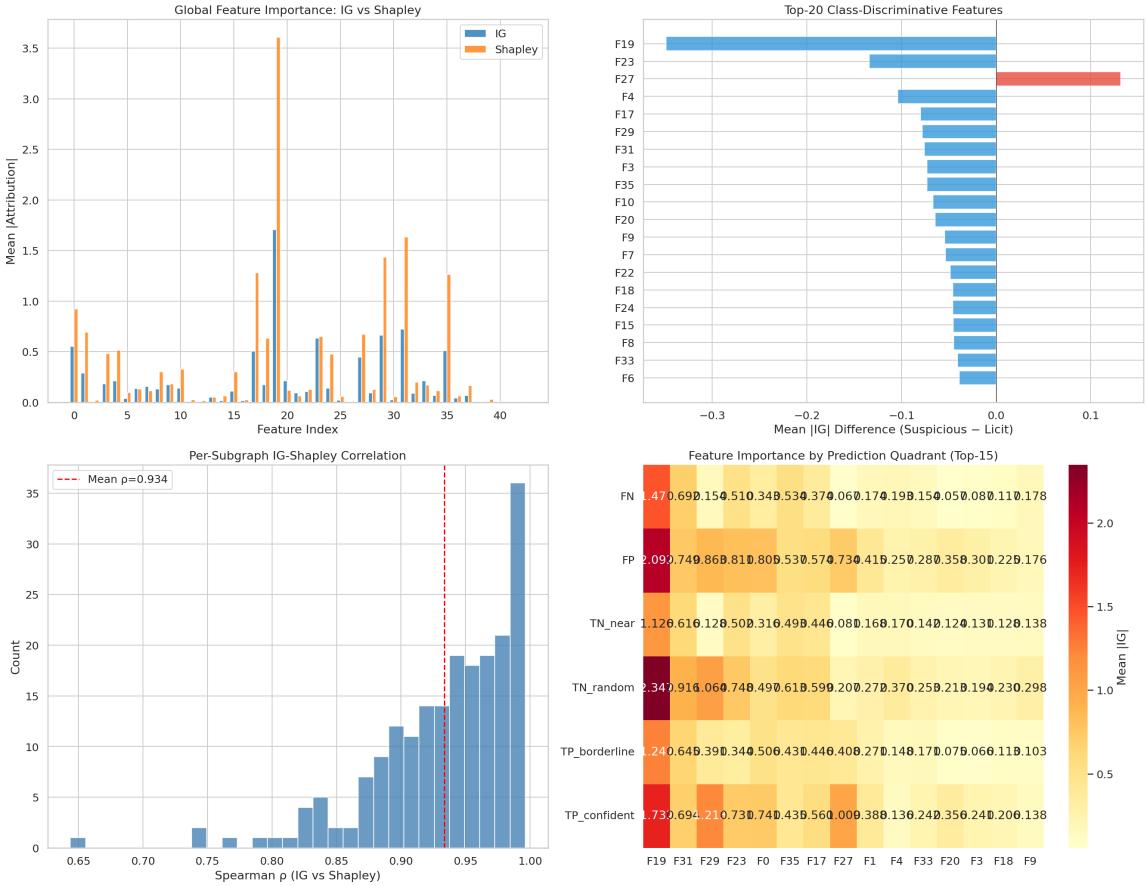

Figure 6.8: Feature attribution analysis on the 200-subgraph stratified sample. Top: global absolute-mean importance rankings for Integrated Gradients and Kernel SHAP; Spearman rank correlation $\rho = 0.9505$. Bottom: signed differences between the mean attribution on correctly classified suspicious and correctly classified licit subgraphs.

contrast is well established in the feature-selection literature (Slack et al., 2020), and the difficulty is particularly pronounced for the Elliptic2 task, in which the positive class is small in volume and explained features do not correspond straightforwardly to the class of interest.

Second, Elliptic2 does not release the feature dictionary, instead providing node features as 43 binned ordinal dimensions without a semantic description, which is done specifically to preserve the intellectual property of the dataset creators (Bellei et al., 2024). While this precludes any claims about the actual transaction-related meaning of the features in question, it does not impinge on the interpretative validity of the attribution results. The results presented here can establish that two feature attribution methods converge, but cannot determine what a given node feature corresponds to (for example, transaction amount, wallet age, or peeling-chain behaviour). This is a valid restriction, not a fatal one: a compliance analyst using the model as a triage tool does value a consistent attribution signal, even absent closed-form feature semantics, as that allows them to check the attributions manually against the raw on-chain data.

6.3.3 Cross-method agreement

The single strongest positive result in this chapter is the agreement between Integrated Gradients and Kernel SHAP. Computing global per-feature importance as the absolute-mean attribution over all 200 subgraphs, and the Spearman rank correlation of the two, gives $\rho = 0.9505$. The direction of the top-five signed differences also agrees: F19 is negative in both cases (IG: -0.349, SHP: -0.490), F27 is positive in both cases (IG: $+0.132$, SHP: $+0.279$), and the remaining top-five features have equal membership though not order. We can care about this because Integrated Gradients and SHAP differ by construction. IG computes a straight-line integration of input-space gradients between a reference baseline and the actual input, and as such it is sensitive to the choice of baseline and to the model's local linearity assumptions. Kernel SHAP computes coalitions of features to be tested and uses a weighted linear regression to approximate the Shapley values, so it is sensitive to the sampling budget and to the choice of background distribution. The convergence of two methods which operate under quite different assumptions on the same feature ranking gives us a strong argument that the signal in the model itself is robust to the choice of attribution method. To argue that both methods are equally wrong would require a great deal of defence on behalf of the methods themselves, given the great methodological distance between the two. What this does not show is that these explanations are correct as they pertain to the true causal structure of the underlying data. Establishing that would require ground-truth feature relevance, which the Elliptic2 dataset does not provide. What it does establish is that two orthogonal attribution methods agree. This is a necessary, if not sufficient, condition for trustworthy feature explanations.

6.4 Structural explainability

6.4.1 Method coverage and the near-uniform metric

We tried four structural explanation methods on our primary model: GNNExplainer (Ying et al., 2019), PGExplainer (Luo et al., 2020), the attention weights of the trained GATv2 layers, and SubgraphX (Yuan et al., 2021). Each generates some degree of per-edge or per-node importance for edges and nodes in a predicted subgraph. To quantify how informative any given explanation is, we compute the per-subgraph importance and then score it using a "near-uniform" threshold: a subgraph mask is flagged near-uniform if its standard deviation is less than 10% of the mask range, which means that the explanation method could not distinguish any specific edge or node from the others. The lower the rate of near-uniform masks, the better; a method that is never near-uniform produces informative explanations on all 200 subgraphs.

Table 6.5 reports the near-uniform rates of each method, stratified by subgraph size. GNNExplainer was informative (non-uniform) on 200 of 200 subgraphs, with attention informative on all 2-edge and 3+-edge subgraphs and forced to be uniform on 1-edge subgraphs due to the lack of additional edges to differentiate between. Consequently, the 67% rate is a degenerate near-uniform performance at the minimum subgraph size, not a breakdown of attention as an explanation method. SubgraphX yields a discrete subset of retained nodes, not a continuous mask, and thus cannot be compared by this metric; we

discuss its behaviour in detail in Section 6.4.3.

Table 6.5: Near-uniform rate of structural explanations, by subgraph edge count. Lower is better. $n = 200$ subgraphs in total.

Method	1-edge ($n = 69$)	2-edge ($n = 61$)	3+-edge ($n = 70$)
GNNExplainer	0%	0%	0%
PGExplainer	100%	100%	100%
Attention (GATv2 layers)	67%	0%	1%
SubgraphX	discrete (n/a)	discrete (n/a)	discrete (n/a)

6.4.2 Fidelity on sparse subgraphs

Fidelity+ and Fidelity- are standard metrics for evaluating structural explanations (Agarwal et al., 2022). Fidelity+ measures how much prediction confidence drops when the top-ranked explanation edges are removed (higher is better: the explanation correctly identifies which edges the prediction relied upon). Fidelity- measures how much confidence drops when the bottom-ranked explanation edges are removed (lower is better: the explanation correctly identifies which edges the prediction did not rely upon).

For GNNExplainer, across the 200-subgraph sample: Fidelity+ $= 0.0066 \pm 0.1422$, Fidelity- $= 0.0195 \pm 0.1391$. The average fidelity scores are near zero. However, their standard deviations are over one magnitude larger than their means, which at first glance looks like poor performance, and a negative result for the explanations. But it is not. It is a limitation of the fidelity metric on sparse subgraphs.

The median Elliptic2 subgraph contains 3 nodes; across the 200-subgraph stratified sample, edge counts range from a single edge in the 2-node strata to a few dozen edges in the 4+-node strata. When a subgraph only has 2 edges, removing the top 50% of the explanation edges (one edge) very likely results in the subgraph being disconnected, creating a subgraph that consists entirely of disconnected nodes (singleton components). At that point the graph neural network would be producing predictions from a structurally collapsed input. The original definition of fidelity presupposes a graph that is dense enough that edge removal can produce a structurally valid counterfactual (or the metric would have to be adjusted). This assumption breaks down when edge count is as low as 2–3. Thus, the high standard deviation across the sample is not necessarily a sign of "bad" explanations, but rather the variance resulting from subgraphs surviving intact after this perturbation versus subgraphs fully collapsing.

A more methodologically robust alternative metric is the entropy of the mask itself, which assesses the peakedness of the explanation without requiring a counterfactual pass through the model. The mean entropy of GNNExplainer masks on this sample is 0.489 bits across 2-node subgraphs, 1.091 across 3-node subgraphs, and 1.926 across 4+-node subgraphs. Entropy grows with subgraph size as one would expect (higher edge counts enable higher entropy), and the absolute value of the entropies indicates that GNNExplainer masks are neither peaked nor flat on average. Taken together with the zero near-uniform rate, the entropy profile reinforces the argument that GNNExplainer generates substantive structural explanations, even if the fidelity metric is not a well-suited tool to evaluate their performance on this task.

The cross-method Jaccard score between GNNExplainer- and attention-predicted edges (top 50%) is 0.41 on average, at 0.46 on 1-edge subgraphs (the degenerate case where Jaccard values can either overlap entirely or not at all), 0.40 on 2-edge subgraphs, and 0.36 on 3+-edge subgraphs. While moderate, this result provides an informative picture: GNNExplainer and attention simply do not agree on which edges are most critical for predictions because, to some extent, these methods answer different questions. Attention identifies edges through which message-passing layers directed information during the forward inference pass; GNNExplainer maximises a sparse mask that matches the model's output. While the goals are aligned, they are not exactly the same, and the 0.36–0.40 Jaccard values on multi-edge subgraphs reflect this partial alignment. SubgraphX works on the node level, not the edge level, and uses Monte Carlo Tree Search to find a substructure that matters. Of 50 subgraphs sampled, the median node retention was 3.0 out of 6.0, with a mean retention ratio of 48% across the sample. Unlike GNNExplainer and attention, SubgraphX provides a binary choice per node, which means lower resolution but is more straightforward to convey to an analyst in an explanation: "the model's suspicious prediction relies on this set of three nodes" is more digestible than a weighted importance score spread across six edges.

6.4.3 A note on PGExplainer

PGExplainer was slated for initial experimentation but is left out of cross-method comparison. Under four training conditions, namely the PyTorch Geometric default setup, the setup that trains for the phenomenon and labels separately, the counterfactual formulation, and the formulation that trains for both factual and counterfactual with labels balanced across the two classes, PGExplainer produced all-zero masks on all 200 of the test subgraphs, which yielded a 100% near-uniform rate at every subgraph size. On the same set of subgraphs, under the same trained model, GNNExplainer produced informative masks in 200 of 200. This is a methodological failure, not a task failure. It corroborates previous findings that PGExplainer is sensitive to class imbalance and subgraph size (Agarwal et al., 2022), both of which are the case for the Elliptic2 problem. We defer a thorough investigation of how PGExplainer can be made to work in this setting to future work.

6.5 Synthesis

All three axes point to the same conclusion. On the predictive performance axis, a feature-aware, subgraph-local GNN outperforms the best previously published structure-only GNN baseline by 2.47× on PR-AUC, a result that generalises to the three architectures we tested. On the feature-attribution axis, two methods that rely on different assumptions correlate at $\rho = 0.9505$ on the global attribute importance and identify the same small set of discriminative attributes, subject to the caveat that the feature dictionary for Elliptic2 is anonymised and that the interpretation of feature attribution should be read as a signed class contrast rather than as an absolute value. On the structural explanation axis, three of four methods (GNNExplainer, attention, and SubgraphX) generate informative explanations on the set of subgraphs we evaluated, subject to the caveat that fidelity as a metric is ill-suited to subgraphs of two to three nodes, and that the near-uniform

rate at the 1-edge size boundary reflects the metric failing rather than a methodological failure of attention. The main takeaway is that the Elliptic2 classification problem relies primarily on node attributes, and that the GNN has found a way to use these attributes in a manner that multiple explanation methods concur on. The structural explanations are an ancillary line of evidence that is less amenable to quantitative evaluation at this subgraph size, but qualitatively consistent with the feature-attribution findings. Chapter 7 discusses ethical considerations and potential biases. Chapter 8 discusses the additional engineering required to deploy this system into production.

7. Ethical and Bias Assessment

We evaluate the performance of the explainable GNN-based AML pipeline against the requirements of the seven criteria defined in the European Commission's Assessment List for Trustworthy Artificial Intelligence (ALTAI) (High-Level Expert Group on Artificial Intelligence, 2020). The structure mirrors the official self-assessment template, but we have filled in each part by referring to our specific technical and ethical design choices. When our framework cannot meet each requirement, we have provided an alternative explanation. Moreover, each blind spot has been identified and explained as best we can, as we could not perform a full check on some ALTAI criteria with the system we have.

7.1 Human agency and oversight

We designed our system to function as a decision-support tool, never as an independent classifier that makes predictions by itself. While the GATv2 model is able to predict suspicious components, we did not make the decision on how to deal with these results. In the real world of an actual AML analyst in a bank, the decision to submit a SAR depends entirely on the compliance expert and their ability to understand the results of the model and the risk that the component poses to the bank. We set our threshold accordingly and made it a high one ($t = 0.913$). Since the precision-recall curve of our model showed that at this threshold our system has a precision of 0.649 and a recall of 0.429, the choice seems sensible. A false positive here has a higher cost for the analyst, since they have to spend more time investigating and the system might produce more false alarms than a true alert, whereas a false negative is recoverable downstream through other controls. Our choice of threshold and the decision-making process were chosen with the goal that all suspicious transactions are put under the human eye and that a lower threshold would only be chosen if the system was fully automated. When the predictions of the model are shown to the human analyst, the explanation allows them to either accept, override, or escalate the result with their own justification.

7.2 Technical robustness and safety

We approached technical robustness from an optimisation perspective, with a data perspective, and an explanation perspective. In terms of optimisation, we selected hyperparameters with the Optuna search tool using a median pruner, running thirty trials per architecture in the primary sweep and a further thirty-five trials in the GATv2 refinement. We did not use ROC-AUC as a performance measure, as the imbalance 43:1 ratio (97.73% licit subgraphs, 2.27% suspicious, Section 3.3), and so PR-AUC (which gives more weight to performance on the rare class) is the more relevant metric, as a good performance on the majority class can result from just learning to predict the majority class. Finally, we trained the model with an early stopping mechanism using a patience window of fifteen epochs, such that we would not overtrain our model on a particular fold. We also took precautionary measures to prevent node leakage on the data by dividing our subgraphs on a component basis, stratifying this process. When we trained our model on this data,

we applied a weighted cross-entropy loss, as the model could trivially achieve high performance on an ROC by simply guessing the licit node. Finally, our approach was to test robustness on an empirical basis, instead of simply being a property. We compared the feature attributions produced from IG and Kernel SHAP through a Spearman rank correlation and got a value of $\rho = 0.9505$. We cross-compare the results from the feature-based attributions and the GNNExplainer results, as these are the most prominent methods for producing explanations on this type of graph. Finally, we cross-checked the results with an edge mask on the edges that were chosen by the explanation. Our aim was to cross-examine different methods to compare and contrast them, rather than simply aggregating results that are highly similar, since the highest cross-comparison is a stronger indication of trustworthiness than an individual method on its own. The models do not address any potential residual errors in labels, as these are commercial determinations by Elliptic, not determinations made by a regulatory authority. If there is any systemic bias in Elliptic's own labelling priors, then that bias will necessarily leak into the model's predictions, but that bias is opaque from the interior of this pipeline.

7.3 Privacy and data governance

Our system is structurally private since Elliptic2 is a dataset with node features and edge features that have already been binarised or ordinalised and that do not consist of personal identifiable information. There are 43 pre-binned node features and 95 pre-binned edge features. We did not add extra data sources to our system that we had to crawl, download, or join. To prevent any form of leakage, we considered it part of our pipeline not to add external data, or to include any private or personally identifiable data in the model. The statistics for normalisation were calculated on the training split only, then reused on validation and test. We saved the splits, the feature stats, and the fingerprints as artefacts so any downstream metric can be linked to the exact data state that was used to calculate it.

7.4 Transparency

Our main investment was in transparency. The project's transparency ceiling is the hardest constraint. However, we can say that we version the whole notebook, we save the artefacts at each stage with checkpoints, we explain in three levels, and we have different explanations that can be combined or interpreted individually: attributions that point to what features are informative, structural attributions pointing to which edges contributed to the prediction, and cross-method agreement (Spearman $\rho = 0.846$ between the global feature importance derived from aggregated GNNExplainer node masks and the Integrated Gradients global feature ranking).

The ceiling is the feature anonymisation of the dataset itself, already highlighted in Section 3.3 as a design limitation. We can tell the model (and, in turn, the researcher) which subgraph gives Feature 35 the biggest contribution value, but that information has been anonymised away. We can never be sure what transaction feature this actually is (for example, it can only be found in Elliptic's feature dictionary). From an application point of view, it means a regulator or supervisor inquiring "Why is this transaction flagged?"

will be answered with a structure of "which features of the transaction caused it to be flagged" (which is complete and fully valid), but the answer "high velocity of transfer" or "small denomination structuring" (which is a business-level semantic description) cannot be given back (which is not complete and valid).

7.5 Diversity, non-discrimination and fairness

We find that fairness is the only dimension that cannot be entirely examined because of the same reason given in Section 3.3: Elliptic2 does not include any demographic, geographic, or institutional attributes. The evaluation of fairness usually requires protected attributes, and they are unavailable in the data; hence, there is no possible way of evaluating whether different groups of users are discriminated against.

We can however test for structural bias, and we found a clear structural bias from examining the explanation sample: of the 40 false positives in the 200-subgraph stratified explanation sample, 28 (70%) are 2-node subgraphs. These 2-node subgraphs have no meaningful aggregation since the nodes are already very minimal in size. Therefore, the GNN essentially uses only node features for these, making the false positive rate much higher in this sample compared to the other samples containing 3-node subgraphs and above. It is not a 'fairness' issue as such, but it is a bias on an operational level: a user whose on-chain activity forms a 2-node component is more likely to be falsely flagged than one that forms a larger, more complex subgraph.

An honest strategy to mitigate this is to apply a larger threshold to 2-node samples or send them to a dedicated model that does not use any graph structure. Regardless of the solution you choose, in a production environment it would be vital to continuously monitor the FP rate by both subgraph size and transaction type (to monitor this bias as it changes over time), as well as by protected attributes (to monitor any unfair discrimination that cannot be uncovered through this anonymised research dataset).

7.6 Environmental and societal well-being

To put things in perspective on an environmental level, every single training run utilised rented cloud GPUs (via Vast.ai), and all training ran on a rental cloud GPU of NVIDIA's 5090 architecture for the duration of four days in February 2026, using approximately 75 total GPU-hours. The final cost of training amounted to approximately EUR 15 (a rough approximation to reflect the exchange rate at the time of writing), and when we account for the published total design power of 575 W for the RTX 5090, that is approximately 43 kWh of electricity. That means that training the models on a system that was using a European electrical grid would produce approximately 14 kg of CO_2, based on the average carbon intensity of 0.328 kg CO_2/kWh used by the European Environment Agency (European Environment Agency, 2023). This is roughly equivalent in scale to driving a small car 80 km. It should be noted that Optuna's median pruning function cut short the majority of failed training runs, as does the nature of the candidate architecture designs (which are all intentionally small, in total, between 95k to 289k parameters), both of which contribute to the training time of each individual run.

On the matter of societal well-being and ethics, the case for explainable AML systems is straightforward and urgent. Money laundering is a necessary component of various predicate crimes, and can include but is not limited to illicit drug trafficking, corruption, and financing of terror networks. However, the cryptocurrency channel is increasingly being used for money laundering and, despite the development of various tools, continues to expand beyond the scope of traditional AML regulations. Consequently, it is essential that the tools developed to detect money laundering not just work, but be explainable. An opaque model making high-stakes financial decisions is inherently problematic because, even when correct, there exists no ground on which the decision can be contested or appealed.

7.7 Accountability

For this thesis, the concept of accountability has been operationalised in terms of reproducibility. By using set random seeds, persisting split assignments and saving every model checkpoint alongside the model's hyperparameter configuration, we have ensured that every metric cited in this thesis can be independently recreated by re-running a notebook on the saved artefacts. This thesis's associated repository is made publicly accessible. In addition, the explainability pipeline we developed is also an audit pipeline. For every prediction the system makes, one can see which features the integrated gradients attribution profile and GNNExplainer edge mask have attributed to the result, and make an evaluative judgement as to whether those features support or contradict the classification. Limitations were addressed head on, and surprising results are explicitly stated in this thesis (notably, the fidelity metric degradation on small subgraphs from Section 6.4.2, as well as the PGExplainer loss collapse). As we see it, an honest accounting of limitations should have the same epistemic weight as an accurate accounting of capabilities.

The results of the seven area assessment can be summarised in Table 7.1.

Table 7.1: Summary of ALTAI self-assessment for the explainable GNN-based AML detection system.

ALTAI Requirement	Applicability	Key Measures
Human agency and oversight	High	Decision-support framing; precision-weighted threshold ($t = 0.913$); per-prediction explanations
Technical robustness and safety	High	Stratified component-level splits; early stopping; weighted loss; multi-method explanation cross-validation
Privacy and data governance	Moderate	Anonymised features; no PII; research licence; leakage-safe preprocessing
Transparency	High	Three-layer explanations; cross-method agreement; semantic ceiling from anonymisation explicitly stated
Diversity and fairness	Limited	Protected-attribute auditing impossible; structural bias on 2-node subgraphs (28/40 FPs); size-stratified threshold proposed
Environmental and societal well-being	Moderate	$\sim$43 kWh / $\sim$14 kg CO_2; pruned search; compact architectures
Accountability	High	Reproducible pipeline; per-prediction attribution and edge-mask reports; open limitation reporting

8. Production

Creating an effective research pipeline with strong metrics on a holdout test set is not the same as building a production-ready system. This chapter charts the way forward from the notebook-based artefacts discussed in Chapters 5 and 6 to a deployable service. We describe this progression in two parts: an existing public demo, and a proposed enterprise architecture that meets the demands of a real financial entity. We then describe the three production checks the system needs to pass upon each release, locate this design within the high-risk classification of the EU AI Act, and provide a realistic cost estimate.

8.1 Migration approach

The research pipeline currently takes the form of a sequence of Jupyter notebooks (Wb01 to Wb05) that ingests raw Elliptic2 files, generates the labelled subgraph universe, trains the GATv2 model, and executes the explanation pipelines. Each stage of this pipeline saves its intermediate artefacts (split assignment, normalisation statistics, model checkpoints, and attribution arrays), making it reproducible but not a service.

The proposed migration to production will be done in two stages. The first stage is a public demo showing that the trained model and the explanation infrastructure can perform end-to-end on standard CPU-only hardware. No GPU is used during inference, and no data remains on the server after inference. This demo is already deployed at https://thesis.neri.wtf. The second stage is a proposed enterprise architecture on Microsoft Azure, capable of supporting streaming transaction data with an auditable per-prediction output at the latency that would satisfy a compliance department. The enterprise architecture presented in Section 8.3 is merely a proposal and not a fully implemented solution; the only operational artefact is the static demonstration from Section 8.2. Two non-organisational challenges have stymied attempts to realise the architecture.

First, streaming windowing is still a problem that remains unsolved for labelled-subgraph classification: until the graph's boundary is established, a subgraph cannot be evaluated, and that boundary is not directly observable in a live Bitcoin stream since there is no packet that declares "this connected component is now complete." Thus, any deployed classifier must adopt a windowing strategy (time-based cutoff, degree stability criterion, external heuristic) and endure either the performance cost of assessing incomplete graphs or the latency cost of delaying predictions; the existing research does not present a consensus on the Elliptic2 approach for determining graph completeness. This does not apply to the runaway-neighbours problem that affects node-level GNN streaming inference: a cutoff for the message-passing is fixed once a `ccId` is assigned, and the only windowing problem remaining is when to assign the `ccId`.

Second, the use of patented features makes the pipeline inoperable for real-world applications in both upstream and downstream applications. With respect to the former, Elliptic's proprietary feature extraction is the process by which raw blockchain transactions are converted into the 43-dimensional node features and 95-dimensional edge features used by Elliptic2 as published and used by this thesis's model. If someone other than Elliptic wished to deploy this model, that party would either need to recreate or replace

this pipeline, which is both a licensing concern and an involved engineering undertaking. On the downstream side, features are obfuscated in such a way that the meaning of the 43 channels cannot be ascertained: some may be sensitive data (geographical inference, exchange associations, or proxies for wallet holders) or correlates thereof. A compliant deployment would be unable to meet the GDPR requirement for a meaningful explanation of a decision under Article 22, or the transparency requirement of the EU AI Act, since the features determining the decision are opaque to the deployer. Note that Section 7.4 raises the same concern from a trustworthy AI perspective: in practice, such constraints present an insurmountable hurdle.

As an architectural target (but not as an end product) this Azure deployment scenario is still described below. It indicates what a compliant deployment would be, subject to the prior resolution of both windowing rules and feature semantics, two matters that occur prior to architectural choices.

8.2 Demonstration deployment

The public demo is hosted at https://thesis.neri.wtf and titled "Explainable AML Detection". The demo is a single-page React application, built using Vite as its module bundler, and styled using the utility-first CSS framework Tailwind CSS. It is hosted on Vercel, which deploys the static bundle via a global CDN to a set of edge nodes. The deployment adds virtually zero maintenance overhead for the thesis author.

This single-page application has six major sections: Dataset, Models, Predict, Explain, Threshold, and About. The first page, titled Dataset, displays the 121,810 labelled subgraphs of the Elliptic2 dataset, along with summary statistics for the dataset itself, such as node and edge counts (49.4 million nodes, 196.2 million edges). The second page, Models, shows the progression of the model architectures that were built for this research project, including the baseline logistic regression and the final GATv2 attention-based graph classification model. This page lists the best results obtained for each of the models tested, including the test performance (PR-AUC), relative GLASS performance improvement over the logistic regression, the Optuna trial limit for that particular model architecture, and the best performance (precision) obtained for each recall level.

The next page, Predict, allows a user to explore a sample of the 200 labelled subgraphs, applying a filter for confusion quadrant (e.g., `TP_confident`, `TN_random`) or subgraph size (2 nodes, 3 nodes, 4 nodes and greater). These filters mimic the biased sampling that we intentionally performed in Chapter 6; furthermore, the 2-node false-positive bias discovered in Chapter 7 will be immediately obvious to a user filtering the sample by these metrics. Finally, clicking the link for a specific subgraph in this page will display that graph as a network diagram, along with the model's predictions and explanation. The prediction and explanation data are precomputed; these are not calculated in real-time in the browser. The computation time required by GNNExplainer for each subgraph is on the order of seconds, and so cannot be recomputed upon each page load without unacceptable latency for an interactive demo.

The next section, Explain, presents a menu to access two different types of graph-level explanations, as discussed in Chapter 5. This section is self-explanatory and will not be elaborated upon here.

The final section, titled Threshold, is where the interactive element of the website is the most pedagogically helpful to the user. The user can adjust a single slider to vary the classification threshold for the model. For each value that is selected, every sample is re-thresholded against its cached score to produce its new binary label, which in turn updates the precision, recall, F1, and flagged count metrics. No model inference runs in the browser; the metrics are derived entirely from a cache of precomputed scores for the holdout test set of subgraphs, which is why these updates are immediate. Three preset buttons are included below the slider to quickly jump to certain common operating points. The user will easily see why the high-precision operating point was selected for the "production" configuration in Chapter 7; by adjusting the slider the user can directly see the effects of these choices rather than just reading in a table what the numbers were.

In the demo, the attention-based model GATv2 that is used for classification has 288,515 parameters, and was trained on the subgraphs which has median size of 3. Since the model is not run at all in the browser, the whole site is a static website bundle, with the 200 stratified subgraphs and their predicted probabilities and explanation artefacts computed once at build time on the workstation and shipped as JSON next to the React bundle, so each request is just a simple request for the static assets.

8.3 Proposed enterprise architecture

An enterprise-grade solution is designed on Azure as a four-tier architecture that adheres to the conventional model of ingest, serve, monitor, and retrain, with additional capabilities to meet the requirements of a regulated AML deployment.

The ingest layer is where transaction event data from the upstream blockchain analytics service arrives via Azure Event Hubs. A Stream Analytics job performs windowing on the transactions and initiates a graph construction stage that generates connected components, materialising them in a graph database (Azure Cosmos DB with its Gremlin API is suitable for `ccId`-keyed reads). The component completion criteria, determining when a subgraph is complete and ready for scoring, also lives in this layer.

The serving layer runs the GATv2 model in a containerised inferencing system on Azure Kubernetes Service, where the images live in Azure Container Registry and are served through an internal load balancer. The same container also hosts the explainability tools (Integrated Gradients for feature importance, as well as a GNNExplainer run either cached or on demand for edge masks), so each request results in both the model prediction and an explanation report.

The monitoring layer captures metrics such as latency, throughput, and prediction distribution in Application Insights; Log Analytics is used to calculate explanation drift (the running Spearman correlation between the current set and the reference feature attribution rankings) and operational fairness metrics (false positive rate, segmented by subgraph size, to track the bias identified in Chapter 7) to feed thresholds for alerts in a triage queue.

The retrain layer is an Azure Machine Learning pipeline. It runs the Optuna search and subsequent full training run on schedule, registers the resulting models in Azure ML, and deploys a model into AKS only after the validation gate in the next section.

8.4 Production validation strategy

Three tests gate a model release. The inference regression test uses a fixed subset of test subgraphs and verifies that the new model's outputs match the output of the old model within some tolerance, to detect silent breakage caused by a runtime upgrade, library upgrade, or loss of precision. The metric benchmark test runs the candidate model on an unseen subset of data; it blocks release if the candidate model's PR-AUC performance drops below some threshold (0.50 is a reasonable choice here based on the experimental results). The explanation consistency test runs Integrated Gradients on a fixed subset of the training data and blocks release if the resulting attribution profiles diverge too much from the previous released version's attribution profiles, to prevent a model from releasing whose prediction accuracy is the same but whose logic is totally different (that is, to detect when a change in model architecture or training dataset has caused conceptual drift that happens to yield the same accuracy).

8.5 Regulatory considerations

In a regulated AML environment, a bank is subject to the requirements of the EU AI Act's high-risk category, specifically Articles 9 to 15, which address risk management, data governance, technical documentation, transparency, human oversight, and accuracy and robustness (European Parliament and Council of the European Union, 2024). The architecture of the present solution directly maps to these requirements. The ALTAI assessment in Chapter 7 is the technical documentation; the per-prediction feature importance and edge attribution reports satisfy the Article 13 transparency requirement at the prediction level; and the decision-support paradigm outlined in Section 7.1 satisfies the Article 14 human-oversight requirement. The reproducibility artefacts outlined in Section 7.7 are sufficient for a conformity assessment and a post-market monitoring requirement. The decision-support paradigm also ensures that the GDPR Article 22 protections against solely automated decision-making are upheld (European Parliament and Council of the European Union, 2016).

8.6 Cost and feasibility

Inference performance is dominated by CPU performance. The model is fast enough to process a single subgraph in a matter of milliseconds on a single core, which puts the per-core throughput of a four-core AKS node at a thousand subgraphs per second, plus headroom for an explanation pass. Running a single core on an Azure D-series machine of comparable specification incurs a small fraction of a cent in cost per thousand classifications, and a two-node minimum HA cluster costs in the hundreds of euros per month. The retraining cost is similar to the cost of the initial training experiment: approximately EUR 15 of GPU time on a cloud GPU marketplace like Vast.ai (or a similarly small fraction of the cost on Azure). In other words, neither the training nor the inference computation is a significant production expense; the significant production expense here is the engineering effort and cost of the graph construction code, the monitoring and triage tools, and the validation gate that stands in front of the deploy step. These are all tractable obstacles

to deploying such a solution.

9. Conclusion, Discussion, and Future Work

9.1 Contributions

This thesis asked whether an explainable graph neural network trained only on node features of labelled Elliptic2 subgraphs can match or exceed structure-based baselines on the AML task, and whether the resulting explanations contain useful signals for compliance analysts. We have developed and assessed such a system and examined it against the ALTAI trustworthy-AI principles (High-Level Expert Group on Artificial Intelligence, 2020) and the EU AI Act (European Parliament and Council of the European Union, 2024). This work leads to four main contributions.

First, we contribute a node-feature based subgraph-local benchmark on Elliptic2 (Bellei et al., 2024). A two-layer GATv2 with global attention pooling achieves a test PR-AUC of 0.515 against the best published structure-only baseline of 0.208, an improvement of approximately 2.47 times. We qualify this: the published baselines used the 196M-edge background graph but no node features; this work used node features but not the background graph. Rather than showing which modelling paradigm is stronger, the comparison points to where the discriminative signal lives in Elliptic2: in node attributes rather than in macro-topology.

Second, we compare two feature-based explainer methods of different lineage, Integrated Gradients and Kernel SHAP, on a stratified 200-subgraph subset of Elliptic2. The rank orders of feature importance generated by the two methods have a Spearman correlation of $\rho = 0.9505$, with sign concordance on the top-five features. This cross-method agreement is not proof of causal correctness, but it is a necessary condition for trustworthy attribution.

Third, we provide a broader empirical pass on structural explainers than is typical at this scale, including GNNExplainer (Ying et al., 2019), GATv2 attention weights, and SubgraphX (Yuan et al., 2021). We show that the fidelity metric breaks on a median three-node subgraph and propose mask entropy as a better diagnostic. We also surface a structural bias: 28 of 40 false positives in the explanation set are two-node subgraphs.

Finally, we assess the pipeline against the ALTAI seven dimensions and the EU AI Act's binding obligations. Our GNN explainers do not yet provide transparency for a non-expert user. They are not demonstrably robust to adversarial graph perturbations (Jin et al., 2021). Fairness on a feature-anonymised dataset cannot be guaranteed, and the EU AI Act's diversity and non-discrimination requirements remain unaddressed by this work.

9.2 Discussion of limitations

The most generous interpretation of our work is constrained by the scope of the Elliptic2 dataset. Our evaluation was limited to a single benchmark dataset; it remains to

be seen whether these results generalise to different blockchains, longer observation windows, or non-Bitcoin anti-money laundering (AML) workflows. The synthetic datasets we considered, AMLworld (Altman et al., 2023) and AMLSim, are useful for analysing how well their generators synthesise realistic patterns, but they do not assess laundering behaviour directly. The comparison in Section 9.1 between node attributes alone and graph structure alone (Bellei et al., 2024) is a partial analysis; a combined model is the natural follow-up. We chose not to optimise our model for a single operating point by adopting Jumping Knowledge concatenation; an alternative implementation with this method yielded a comparable PR-AUC at higher recall, which could be preferable for a production setting in which a false negative outweighs the analyst burden of a false positive. We name the trade-off rather than absorb it into a single metric. Variance on sparse subgraphs renders individual feature importance results suggestive rather than authoritative, and the absence of a feature dictionary means our discriminative channels (F23, F27, F35, F19, F29 by signed difference) cannot be mapped to specific laundering activity. Fairness, finally, is unauditable on a feature-anonymised dataset; the EU AI Act's diversity and non-discrimination requirements remain unaddressed, a constraint of the dataset rather than the method.

9.3 Future work

The most obvious next step is to combine all node attributes with the 196M-edge background graph by applying the GLASS labelling technique (Wang and Zhang, 2022) to the attention-pooled GATv2 architecture. If the background graph carries even a fraction of the signal that node features carry, the combined model should exceed the 2.47× multiplier reported here; if it does not, the case that the discriminative signal is concentrated in node attributes is strengthened.

In addition to this work, there are at least four other avenues worth exploring:

- transfer learning of our attention-pooled network to a larger-scale dataset such as Elliptic++ (Elmougy and Liu, 2023) or AMLworld (Altman et al., 2023); it would then be possible to determine if our model's ability to identify laundering is dataset-specific or generalises to real subgraphs of other blockchains;
- counterfactual explainers such as CF-GNNExplainer (Lucic et al., 2022) and their utility in supporting a real risk-based AML approach (Financial Action Task Force, 2025);
- a systematic study of robustness to adversarial graph attacks (Jin et al., 2021), which is now a requirement for certain systems under the EU AI Act (European Parliament and Council of the European Union, 2024) (Article 15) and is largely absent from the AML GNN literature;
- an in-person trial with compliance officers in the real world that would allow us to directly observe whether our generated explanations are useful. No amount of work done in academia will ever be a substitute for that kind of evidence.

9.4 Closing remarks

Returning to the research question: an explainable graph neural network trained only on node features of labelled Elliptic2 subgraphs matches and exceeds the published structure-based baseline, by approximately 2.47× on PR-AUC, with the regime caveat we have repeated throughout. The resulting explanations are stable enough on the feature side to be useful for analyst triage, and informative enough on the structural side to identify the nodes that carried a prediction; they are not strong enough to substitute for human oversight, and the ALTAI gap remains visible. The honest answer is that this work raises the trustworthy-AI bar for AML graph methods modestly, on one problem, with explainability methods that do better than some have said and worse than others have expected. We have demonstrated that the Elliptic2 problem can be tackled with explainable graph neural networks; whether such methods can be trusted in production-grade compliance is a question raised, and sharpened, by this thesis, but not answered.

A. Full Optuna search spaces

This appendix provides the search spaces and best hyperparameter configurations for the three sub-phases of the model development study (Section 5.7) using Optuna's TPE sampler and a median pruner (`n_startup_trials=5`, `n_warmup_steps=10`) across all 3 sub-phases. The random seed for all trials was fixed to 7 and the deterministic CUDA settings in Section 5.5 were enabled.

A.1 Phase Wb03: primary architecture sweep

The Wb03 search explored 3 architectures (GraphSAGE, GCN, and GATv2) in parallel using the same search space (Table A.1). A total of 30 trials per architecture, i.e. 90 trials total, were drawn and roughly 10 trials per architecture survived median-pruning. Results are shown in Table A.2 and include the best configuration per architecture.

Table A.1: Wb03 hyperparameter search space. Thirty trials per architecture.

Parameter	**Range**	**Type**
`arch`	{GCN, SAGE, GATv2}	Categorical (separate study per arch)
`hidden_dim`	{64, 128, 256}	Categorical
`num_layers`	{2, 3}	Categorical
`dropout`	$[0, 0.5]$	Float, step 0.05
`lr`	$[10^{-4},\ 5 \times 10^{-3}]$	Log-uniform
`pool`	{mean, max}	Categorical
`heads`	{1, 2, 4}	Categorical (GATv2 only)
Fixed: batch size 256, max 80 epochs, patience 15, seed 7		

Table A.2: Wb03 best configurations per architecture, selected by validation PR-AUC. Threshold is the validation-set F1-optimal value transferred unchanged to the test split.

Arch	**hidden**	**layers**	**dropout**	**lr**	**pool**	**heads**	**Val PR-AUC**	**Threshold**
GraphSAGE	128	3	0.10	1.57×10^{-3}	max	—	0.5328	0.9597
GCN	256	2	0.15	1.65×10^{-3}	max	—	0.4757	0.9265
GATv2	256	2	0.20	7.54×10^{-4}	max	1	**0.5425**	0.8775

Across all three primary-model architectures, max pooling was selected in place of mean pooling for the median three-node Elliptic2 subgraph. This suggests that the node with the maximum activation across the subgraph provides the most signal for classification. Furthermore, it is notable that mean pooling was not selected as the pooling mode for any architecture-restricted sub-study.

A.2 Phase Wb03b: GATv2 refinement

The Wb03b GATv2-only refinement introduced additional levers in the form of Jumping Knowledge concatenation and attention pooling. A total of 35 trials were drawn using TPE (Table A.3) in a more refined search space with respect to the previous, more general Wb03 search space with the objective of zooming in on better results. Finally, 4 explicit ablation configurations were retrained using the strict checkpointing to isolate each specific lever's effect in Table A.4 (the attention-pool retrain is the canonical primary-model configuration at the top of Table A.4).

Table A.3: Wb03b hyperparameter search space. Thirty-five trials total, GATv2 only.

Parameter	**Range**	**Notes**
`hidden_dim`	{128, 256}	64 dropped (never competitive in Wb03)
`num_layers`	{2, 3}	
`dropout`	$[0.05, 0.30]$, step 0.05	Narrowed from $[0, 0.5]$
`lr`	$[5 \times 10^{-4},\ 3 \times 10^{-3}]$	Log-uniform; narrowed from $[10^{-4},\ 5 \times 10^{-3}]$
`pool`	{max, attention}	New: gated attention pooling
`jk_mode`	{none, cat, max}	New: Jumping Knowledge
`heads`	{1, 2}	
`lr_scheduler`	{none, cosine}	New

Table A.4: Wb03b ablation results on the test split, all four corners of the (`pool`, `jk_mode`) cross. The attention-pool row is the canonical retrain checkpoint used throughout the explainability work; the remaining three rows are the original Wb03b ablation pass.

Variant	**Pool**	**JK**	**Val PR-AUC**	**Test PR-AUC**	**Test F1**	**Params**
Wb03 reproduction	max	none	0.5429	0.4887	0.4917	222,466
Attention pool (canonical retrain)	attention	none	**0.5421**	**0.5148**	**0.5163**	288,515
JK only	max	cat	0.5245	0.5157	0.5041	288,002
Attention + JK	attention	cat	0.5398	0.4963	0.5060	551,171

A.3 Phase Wb03c: edge-feature integration

The objective of Phase Wb03c is to determine whether the 95 edge features contain information that is independent from the information already provided by the 43 node features. To probe this question, three stages were conducted:

- Wb03c1, which produced the 95 edge features as described in Section 4.3,
- Wb03c2, which ran a search that produced NNConv and triggered the pre-registered stopping rule with a best validation PR-AUC score of 0.4063 after 5 trials, and
- Wb03c3, which pivoted to GINEConv and TransformerConv after failing NNConv, both of which attach edge information to a node convolution without incurring a parameter blow-up of $\mathcal{O}(d_e \cdot d^2)$.

These three sub-phases are summarised in the following tables (Table A.5 and A.6).

Table A.5: Wb03c2 NNConv search space (stopped after five trials).

Parameter	**Range**	**Notes**
`node_hidden_dim`	{64, 128, 256}	Hidden dimension of node MLPs
`edge_hidden_dim`	{64, 128}	Edge network hidden dimension
`num_layers`	{1, 2}	Stacked NNConv layers
`dropout`	$[0.05, 0.30]$, step 0.05	
`lr`	$[5 \times 10^{-4},\ 3 \times 10^{-3}]$	Log-uniform
`pool`	{max, attention}	
Stopping rule: terminate if no trial reaches val PR-AUC ≥ 0.50 early in the search. *Outcome*: best of five completed trials reached val PR-AUC 0.4063; rule fired.		

Table A.6: Wb03c3 search space, applied identically to GINEConv (30 trials) and TransformerConv (23 trials).

Parameter	**Range**	**Notes**
`hidden_dim`	{128, 256}	
`num_layers`	{2, 3}	
`dropout`	$[0.05, 0.30]$, step 0.05	
`lr`	$[5 \times 10^{-4},\ 3 \times 10^{-3}]$	Log-uniform
`pool`	{max, attention}	
`jk_mode`	{none, cat}	
`heads`	{1, 2, 4}	TransformerConv only

Table A.7: Wb03c headline outcomes against the Wb03b GATv2 + attention pool primary model.

Model	**Trials**	**Val PR-AUC**	**Test PR-AUC**
NNConv (Wb03c2)	5	0.4063	— (stopped)
TransformerConv (Wb03c3)	23	0.4307	— (below threshold)
GINEConv (Wb03c3)	30	0.5393	0.5087
GATv2 + attention pool (Wb03b, reference)	35	0.5421	0.5148

GINEConv is comparable with respect to the Wb03b primary-model in terms of both validation and test PR-AUC, but it does not provide a GATv2-style attention-coefficient readout that serves as a model-native explanation channel as described in Section 6.4. For these reasons, the final primary-model decision retained Wb03b GATv2 + attention pool (Section 5.7).

B. Complete confusion matrices

This appendix lists test-split confusion matrices of all models in this thesis at the validation-set F1-optimal threshold, and the operational metrics calculated from them. The licit subgraph count in the test split is 23,809, the full test split has 24,362 components. The false-positive rate is against the licit population, the alert rate against the split in whole.

Table B.1: Test-set confusion matrices and operational metrics for every model in this thesis. The threshold column is the F1-optimal value chosen on the validation split. The GATv2 + attention pool row reports the canonical retrain checkpoint used in Sections 6.3–6.4; the remaining Wb03b rows report the original ablation pass with no retraining.

Phase	Model	TN	FP	FN	TP	Threshold	FPR	Alert rate
Wb02	LogReg (pooled)	22,920	889	346	207	0.8652	3.73%	4.50%
Wb02	GraphSAGE (default)	23,392	417	300	253	0.9027	1.75%	2.75%
Wb03	GraphSAGE (tuned)	23,666	143	338	215	0.9597	0.60%	1.47%
Wb03	GCN (tuned)	23,603	206	350	203	0.9265	0.87%	1.68%
Wb03	GATv2 (tuned)	23,641	168	323	230	0.8775	0.71%	1.63%
Wb03b	GATv2 + max pool	23,684	125	332	221	0.9176	0.53%	1.42%
Wb03b	**GATv2 + attention pool** (retrain)	**23,681**	128	316	**237**	0.9128	0.54%	1.50%
Wb03b	GATv2 + JK(cat)	23,626	183	305	248	0.9023	0.77%	1.77%
Wb03b	GATv2 + attention + JK(cat)	23,671	138	319	234	0.9247	0.58%	1.53%
Wb03c3	GINEConv (best)	—	—	—	—	—	—	—

The Wb03c3 GINEConv row has no confusion matrix in this appendix since we didn't run the explainability pipeline (Sections 6.3–6.4) against it, the only reported metric on this row is its test PR-AUC of 0.5087 (Appendix A, Table A.7).

Two notable observations. First, all tuned GNNs decrease the count of false-positives, and the alert rate, against the pooled logistic-regression baseline, by at least a factor of 2, with very little fluctuation in true-positive count. Second, the Wb03b 4 ablation variants balance precision vs recall on a well-behaved Pareto front: max pool achieves the highest precision (highest TN, lowest FP), JK(cat) achieves the highest recall (highest TP, lowest FN), and the retrained attention-pool canonical version is on the good operating zone at 0.6493 precision, 0.4286 recall.

C. Fidelity distributions by quadrant and size

This appendix describes the GNNExplainer Fidelity+ and Fidelity- distributions on the 200-subgraph stratified explanation sample. Fidelity scores are listed both by size and by quadrant. The main overall values of Fidelity+ $= 0.0066 \pm 0.1422$ and Fidelity- $= 0.0195 \pm 0.1391$ in Section 6.4.2 cover up some underlying structure shown in the per-stratum tables.

Table C.1: GNNExplainer Fidelity+ and Fidelity- on the 200-subgraph stratified sample, stratified by component size. Standard deviations exceed the means at every size, which Section 6.4.2 attributes to the structural collapse that follows edge removal on subgraphs of two to three nodes.

Size bucket	n	**Fidelity+ (mean ± std)**	**Fidelity- (mean ± std)**
2-node	69	0.0191 ± 0.1521	0.0194 ± 0.1752
3-node	64	0.0009 ± 0.1806	0.0206 ± 0.1284
4+-node	67	-0.0009 ± 0.0756	0.0184 ± 0.1049
Overall	**200**	0.0066 ± 0.1422	0.0195 ± 0.1391

Table C.2: GNNExplainer Fidelity+ and Fidelity- on the 200-subgraph stratified sample, stratified by prediction quadrant. The TP_confident and TP_borderline rows show the smallest standard deviations, reflecting that the model's high-confidence positives carry the cleanest explanation signal.

Quadrant	n	**Fidelity+ (mean ± std)**	**Fidelity- (mean ± std)**
TP_confident	40	0.0065 ± 0.0284	0.0030 ± 0.0152
TP_borderline	20	0.0044 ± 0.0220	0.0706 ± 0.1295
FP	40	0.0506 ± 0.1830	0.0363 ± 0.1172
FN	40	-0.0484 ± 0.1282	-0.0112 ± 0.1468
TN_near	30	0.0700 ± 0.1654	0.0790 ± 0.2185
TN_random	30	-0.0407 ± 0.1717	-0.0338 ± 0.1258
Overall	**200**	0.0066 ± 0.1422	0.0195 ± 0.1391

Table C.3: GNNExplainer mask entropy by subgraph size, repeated from Section 6.4.2 for completeness. Entropy is the proposed mask-only diagnostic that does not require a counterfactual pass through the model and therefore does not collapse on small subgraphs.

Size bucket	n	**Entropy (mean ± std, bits)**
2-node	69	0.4888 ± 0.2099
3-node	64	1.0913 ± 0.2340
4+-node	67	1.9264 ± 0.3344

PGExplainer has been omitted because all 200 masks became zero under all of the training conditions we tried (Section 6.4.3), so fidelity values with regard to a mask that consists entirely of zeros cannot be interpreted. SubgraphX has also been left out because it is a discrete subset of nodes rather than a continuous mask. SubgraphX's compactness is covered in Section 6.4.2 and, where appropriate, on the 50-subgraph subset per-quadrant.

D. Code repository guide

This appendix describes the layout of the accompanying code repository, the order in which the notebooks must be executed to reproduce the results in this thesis, and the artefacts each notebook produces.

D.1 Repository layout

The repository root contains:

- `wb01_e2_preprocessing.ipynb` through `wb04_05_06_explainability.ipynb`: one notebook per phase of the study, executed in numerical order.
- `requirements.txt`: pinned Python dependencies, with PyTorch installed separately to match the CUDA version of the target machine.
- `README.md`: top-level repository description, environment setup notes, and a flowchart of the data pipeline.
- `results/`: per-phase directories (`wb02/`, `wb03/`, `wb03b/`, `wb03c1/`, `wb04/`, `wb05/`, `wb06/`) holding JSON metric files, CSV search results, model checkpoints, and figures.
- `docs/`: supporting reference documents including the explainability statistics writeup, the model development reference, and the canonical context pack.
- `EXPORTS/`: HTML and PDF exports of selected notebooks for offline review.
- `scripts/`: standalone Python scripts for one-off computations such as background-edge statistics.

The Elliptic2 raw data is not redistributed with the repository. The five primary CSVs (`background_nodes.csv`, `background_edges.csv`, `nodes.csv`, `edges.csv`, `connected_components.csv`) must be downloaded separately from the dataset's release page and placed in a directory whose path is configured at the top of `wb01`.

D.2 Environment

Experiments require Python 3.11, PyTorch 2.9.1 with a CUDA build matching the host (the primary model was trained on an NVIDIA RTX 5090 with CUDA 12.8), and PyTorch Geometric. The remaining dependencies are pinned in `requirements.txt`; the recommended installation order is to install PyTorch first via the official wheel index, then `pip install -r requirements.txt`. CPU-only inference is supported and is the configuration used for the public demo at https://thesis.neri.wtf; CPU training is not recommended.

All notebooks set the random seed to 7 (preprocessing uses 42), enable `torch.backends.cudnn.deterministic = True`, disable `torch.backends.cudnn.benchmark`, and set `CUBLAS_WORKSPACE_CONFIG=:4096:8` before the first PyTorch operation. Running the same code on the same hardware produces bit-identical results (Section 5.5).

D.3 Notebook execution order

The notebooks form a directed acyclic graph; later notebooks read the artefacts produced by earlier ones rather than recomputing them. The execution order is:

1. `wb01_e2_preprocessing.ipynb`: reads the five raw CSVs, extracts the labelled universe, builds the 70/10/20 stratified split, and writes `edge_index.npy`, `node_features.npy`, `node_components.npy`, `splits.json`, and `feature_stats_train.json` to the processed-data directory.

2. `wb02_e2_baselinesAndGraphs.ipynb`: trains the pooled logistic regression and default GraphSAGE baselines, writes `baseline_tabular_metrics.json` and `baseline_gnn_metrics.json` to `results/`.

3. `wb03_model_development.ipynb`: runs the Wb03 architecture sweep (30 trials per architecture for GraphSAGE, GCN, GATv2), writes `search_results.json`, `best_hyperparameters.csv`, `model_comparison.csv`, `primary_model_selection.json`, and per-architecture model checkpoints to `results/wb03/`.

4. `wb03b_model_refinement.ipynb` followed by `wb03b_retrain_attention.ipynb`: runs the Wb03b 35-trial refinement search and the strict-checkpoint retraining of the chosen attention-pool configuration. Outputs land in `results/wb03b/` as `search_results.json`, `ablation_results.json`, `attention_only_result.json`, and the canonical primary-model checkpoint `attention_only_state.pt`.

5. `wb03c1_e2_preprocessing_edge_feat.ipynb`: streams the 196.2M-row background edge file with predicate pushdown, aligns edge features to the PyG packed ordering, and writes `edge_features.npy` along with the edge-feature diagnostics figure.

6. `wb03c2_e2_edge_featrures_conv.ipynb`: the NNConv stage, which terminates after five trials per the pre-registered stopping rule.

7. `wb03c3_e2_GINE_Transformer.ipynb` (or the `.py` script of the same name): runs the GINEConv (30 trials) and TransformerConv (23 trials) studies and writes their search-result JSONs.

8. `wb04_05_06_explainability.ipynb`: the explainability stack. Tier 0 produces the 200-subgraph stratified sample written to `results/wb04/explanation_sample.json`. Tier 1 runs Integrated Gradients and Kernel SHAP and writes the per-subgraph attribution arrays to `results/wb05/`. Tier 2 runs GNNExplainer, PGExplainer (which collapses), GATv2 attention extraction, and SubgraphX (50 subgraphs only), writing their

per-subgraph results as pickles to `results/wb06/`. The notebook also computes the cross-method analysis JSONs `wb04_05_06_analysis_summary.json` and `wb04_05_06_analysis_summary_v2.json`; the v2 file uses the absolute-mean ranking methodology described in Section 6.3.3 and is the file from which all explainability numbers in this thesis are drawn.

D.4 Reproducing a specific number

For any quantitative claim in this thesis, the canonical reproduction path is to (a) confirm the input fingerprint by recomputing the SHA256 over the five raw CSVs and comparing against the values written by `wb01`; (b) re-run the relevant notebook with the same seed; and (c) compare the resulting JSON or CSV against the file shipped under `results/`. Because the deterministic-CUDA settings are enabled and seeds are fixed, any divergence at this stage is a sign of an environment difference rather than a stochastic disagreement.

Bibliography

Agarwal, C., Queen, O., Lakkaraju, H., and Zitnik, M., 2022. Evaluating explainability for graph neural networks. *Advances in neural information processing systems 35 (neurips) datasets and benchmarks track* [Online]. arXiv: 2208.09339. Available from: https://arxiv.org/abs/2208.09339.

Akiba, T., Sano, S., Yanase, T., Ohta, T., and Koyama, M., 2019. Optuna: a next-generation hyperparameter optimization framework. *Proceedings of the 25th acm sigkdd international conference on knowledge discovery & data mining (kdd)* [Online], pp.2623–2631. arXiv: 1907.10902. Available from: https://doi.org/10.1145/3292500.3330701.

Altman, E., Blanuša, J., Niederhäusern, L. von, Egressy, B., Anghel, A., and Atasu, K., 2023. Realistic synthetic financial transactions for anti-money laundering models. *Advances in neural information processing systems 36 (neurips) datasets and benchmarks track* [Online]. arXiv: 2306.16424. Available from: https://arxiv.org/abs/2306.16424.

Bellei, C., Xu, M., Phillips, R., Robinson, T., Weber, M., Kaler, T., Leiserson, C.E., Arvind, and Chen, J., 2024. *The shape of money laundering: subgraph representation learning on the blockchain with the Elliptic2 dataset* [Online]. Preprint arXiv:2404.19109. arXiv: 2404.19109. Available from: https://arxiv.org/abs/2404.19109.

Bergstra, J., Bardenet, R., Bengio, Y., and Kégl, B., 2011. Algorithms for hyper-parameter optimization. *Advances in neural information processing systems 24 (neurips)* [Online], pp.2546–2554. Available from: https://papers.nips.cc/paper_files/paper/2011/hash/86e8f7ab32cfd12577bc2619bc635690-Abstract.html.

Brody, S., Alon, U., and Yahav, E., 2022. How attentive are graph attention networks? *Proceedings of the 10th international conference on learning representations (iclr)* [Online]. arXiv: 2105.14491. Available from: https://arxiv.org/abs/2105.14491.

Chainalysis, 2024. *The 2024 crypto crime report*. https://www.chainalysis.com/wp-content/uploads/2024/06/the-2024-crypto-crime-report-release.pdf. Annual industry report on illicit cryptocurrency activity covering 2023 data. Accessed 2026-04. Chainalysis Inc.

Chawla, N.V., Bowyer, K.W., Hall, L.O., and Kegelmeyer, W.P., 2002. SMOTE: synthetic minority over-sampling technique. *Journal of artificial intelligence research* [Online], 16, pp.321–357. Available from: https://doi.org/10.1613/jair.953.

Elmougy, Y. and Liu, L., 2023. Demystifying fraudulent transactions and illicit nodes in the Bitcoin network for financial forensics. *Kdd workshop on machine learning in finance* [Online]. Dataset and code: https://github.com/git-disl/EllipticPlusPlus. arXiv: 2306.06108. Available from: https://arxiv.org/abs/2306.06108.

European Environment Agency, 2023. *Greenhouse gas emission intensity of electricity generation in Europe*. https://www.eea.europa.eu/en/analysis/indicators/greenhouse-gas-emission-intensity-of-1. Accessed 2026-04. European Environment Agency.

European Parliament and Council of the European Union, 2016. *Regulation (EU) 2016/679 of the European Parliament and of the Council of 27 April 2016 on the protection of natural persons with regard to the processing of personal data (General Data Protection Regulation)*. https://eur-lex.europa.eu/eli/reg/2016/679/oj. Accessed 2026-04. Official Journal of the European Union.

European Parliament and Council of the European Union, 2024. *Regulation (EU) 2024/1689 of the European Parliament and of the Council of 13 June 2024 laying down harmonised rules on artificial intelligence (Artificial Intelligence Act)*. https://eur-lex.europa.eu/eli/reg/2024/1689/oj. Accessed 2026-04. Official Journal of the European Union.

Financial Action Task Force, 2025. *FATF annual report 2024–2025: another year of action*. https://www.fatf-gafi.org/en/publications/Fatfgeneral/Fatf-annual-report-2024-2025.html. Foreword by FATF President Elisa de Anda Madrazo; covers July 2024 to June 2025. Accessed 2026-04. Paris: FATF/OECD.

Gilmer, J., Schoenholz, S.S., Riley, P.F., Vinyals, O., and Dahl, G.E., 2017. Neural message passing for quantum chemistry. *Proceedings of the 34th international conference on machine learning (icml)* [Online], pp.1263–1272. arXiv: 1704.01212. Available from: https://arxiv.org/abs/1704.01212.

Hamilton, W.L., Ying, R., and Leskovec, J., 2017. Inductive representation learning on large graphs. *Advances in neural information processing systems 30 (neurips)* [Online], pp.1024–1034. arXiv: 1706.02216. Available from: https://arxiv.org/abs/1706.02216.

High-Level Expert Group on Artificial Intelligence, 2020. *Assessment list for trustworthy artificial intelligence (ALTAI) for self-assessment*. https://digital-strategy.ec.europa.eu/en/library/assessment-list-trustworthy-artificial-intelligence-altai-self-assessment. Accessed 2026-04. European Commission.

Hu, W., Liu, B., Gomes, J., Zitnik, M., Liang, P., Pande, V., and Leskovec, J., 2020. Strategies for pre-training graph neural networks. *International conference on learning representations (iclr)*.

Hutter, F., Hoos, H., and Leyton-Brown, K., 2014. An efficient approach for assessing hyperparameter importance. *Proceedings of the 31st international conference on machine learning (icml)* [Online], pp.754–762. Available from: https://proceedings.mlr.press/v32/hutter14.html.

Jin, W., Li, Y., Xu, H., Wang, Y., Ji, S., Aggarwal, C., and Tang, J., 2021. Adversarial attacks and defenses on graphs. *ACM SIGKDD explorations newsletter* [Online], 22(2), pp.19–34. Available from: https://arxiv.org/abs/2003.00653.

Kingma, D.P. and Ba, J., 2015. Adam: a method for stochastic optimization. *Proceedings of the 3rd international conference on learning representations (iclr)* [Online]. arXiv: 1412.6980. Available from: https://arxiv.org/abs/1412.6980.

Kipf, T.N. and Welling, M., 2017. Semi-supervised classification with graph convolutional networks. *Proceedings of the 5th international conference on learning representations (iclr)* [Online]. arXiv: 1609.02907. Available from: https://arxiv.org/abs/1609.02907.

Li, Y., Tarlow, D., Brockschmidt, M., and Zemel, R., 2016. Gated graph sequence neural networks. *Proceedings of the 4th international conference on learning representations (iclr)* [Online]. arXiv: 1511.05493. Available from: https://arxiv.org/abs/1511.05493.

Lin, T.-Y., Goyal, P., Girshick, R., He, K., and Dollár, P., 2017. Focal loss for dense object detection. *Proceedings of the ieee international conference on computer vision (iccv)* [Online], pp.2999–3007. arXiv: 1708.02002. Available from: https://openaccess.thecvf.com/content_iccv_2017/html/Lin_Focal_Loss_for_ICCV_2017_paper.html.

Lucic, A., Hoeve, M. ter, Tolomei, G., Rijke, M. de, and Silvestri, F., 2022. CF-GNNExplainer: counterfactual explanations for graph neural networks. *Proceedings of the 25th international conference on artificial intelligence and statistics (AISTATS)* [Online]. arXiv: 2102.03322. Available from: https://arxiv.org/abs/2102.03322.

Lundberg, S.M. and Lee, S.-I., 2017. A unified approach to interpreting model predictions. *Advances in neural information processing systems 30 (neurips)* [Online], pp.4765–4774. arXiv: 1705.07874. Available from: https://arxiv.org/abs/1705.07874.

Luo, D., Cheng, W., Xu, D., Yu, W., Zong, B., Chen, H., and Zhang, X., 2020. Parameterized explainer for graph neural network. *Advances in neural information processing systems 33 (neurips)* [Online]. arXiv: 2011.04573. Available from: https://arxiv.org/abs/2011.04573.

Ribeiro, M.T., Singh, S., and Guestrin, C., 2016. "Why should I trust you?": explaining the predictions of any classifier. *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining (KDD)* [Online], pp.1135–1144. arXiv: 1602.04938. Available from: https://arxiv.org/abs/1602.04938.

Shi, Y., Huang, Z., Feng, S., Zhong, H., Wang, W., and Sun, Y., 2021. Masked label prediction: unified message passing model for semi-supervised classification. *International joint conference on artificial intelligence (ijcai)*.

Slack, D., Hilgard, S., Jia, E., Singh, S., and Lakkaraju, H., 2020. Fooling LIME and SHAP: adversarial attacks on post hoc explanation methods. *Proceedings of the aaai/acm conference on ai, ethics, and society (aies)* [Online]. arXiv: 1911.02508. Available from: https://arxiv.org/abs/1911.02508.

Sundararajan, M., Taly, A., and Yan, Q., 2017. Axiomatic attribution for deep networks. *Proceedings of the 34th international conference on machine learning (icml)* [Online], pp.3319–3328. arXiv: 1703.01365. Available from: https://arxiv.org/abs/1703.01365.

Suzumura, T. and Kanezashi, H., 2018. *AMLSim: a multi-agent simulator for anti-money laundering*. https://github.com/IBM/AMLSim. Accessed 2026-04. IBM Research.

United Nations Office on Drugs and Crime, 2024. *World drug report 2024: key findings and conclusions*. https://www.unodc.org/unodc/en/data-and-analysis/world-drug-report-2024.html. United Nations publication Sales No. E.24.XI.5. Accessed 2026-04. Vienna: United Nations Office on Drugs and Crime.

Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., and Bengio, Y., 2018. Graph attention networks. *Proceedings of the 6th international conference on learning representations (iclr)* [Online]. arXiv: 1710.10903. Available from: https://arxiv.org/abs/1710.10903.

Wang, X. and Zhang, M., 2022. GLASS: GNN with labeling tricks for subgraph representation learning. *Proceedings of the 10th international conference on learning representations (ICLR)* [Online]. Available from: https://openreview.net/forum?id=XLxhEjKNbXj.

Weber, M., Domeniconi, G., Chen, J., Weidele, D.K.I., Bellei, C., Robinson, T., and Leiserson, C.E., 2019. Anti-money laundering in Bitcoin: experimenting with graph convolutional networks for financial forensics. *Kdd workshop on anomaly detection in finance* [Online]. arXiv: 1908.02591. Available from: https://arxiv.org/abs/1908.02591.

Xu, K., Li, C., Tian, Y., Sonobe, T., Kawarabayashi, K.-i., and Jegelka, S., 2018. Representation learning on graphs with jumping knowledge networks. *Proceedings of the 35th international conference on machine learning (icml)* [Online], pp.5453–5462. arXiv: 1806.03536. Available from: https://arxiv.org/abs/1806.03536.

Ying, Z., Bourgeois, D., You, J., Zitnik, M., and Leskovec, J., 2019. GNNExplainer: generating explanations for graph neural networks. *Advances in neural information processing systems 32 (neurips)* [Online], pp.9240–9251. arXiv: 1903.03894. Available from: https://arxiv.org/abs/1903.03894.

Yuan, H., Yu, H., Gui, S., and Ji, S., 2023. Explainability in graph neural networks: a taxonomic survey. *Ieee transactions on pattern analysis and machine intelligence* [Online], 45(5), pp.5782–5799. arXiv: 2012.15445. Available from: https://doi.org/10.1109/TPAMI.2022.3204236.

Yuan, H., Yu, H., Wang, J., Li, K., and Ji, S., 2021. On explainability of graph neural networks via subgraph explorations. *Proceedings of the 38th international conference on machine learning (icml)* [Online], pp.12241–12252. arXiv: 2102.05152. Available from: https://arxiv.org/abs/2102.05152.