Faculty of Business Administration and Economics

Professorship in Financial Economics and Capital Market Theory

Thesis to obtain the academic degree of Master of Science

Attention-Based Dynamic Multilayer Graph Neural Networks for Credit Risk Modeling

Udval Oyunsaikhan

European University Viadrina

Supervisor: Prof. Dr. Sven Husmann

Second Supervisor: Dr. Rick Steinert

Study Program: MSc in International Business Administration – Data Science and Decision Support

Matriculation Number: 148181

Date: 02 January 2026

Signature of the Author

Abstract

Graph Neural Networks (GNNs) represent the state of the art in credit risk modeling, integrating borrower connectivity with temporal dynamics. However, existing frameworks, such as the DYMGNN proposed by Zandi et al. (2025), rely on computationally intensive full-clique topologies and rigid metadata, which limits their deployability in resource-constrained environments. This thesis addresses these limitations by investigating scalable, behavior-driven topological alternatives on high-end consumer-grade hardware.

The study proceeds in two methodological phases. First, a Constraint-Aware Replication (Hypothesis 1) evaluates whether the predictive logic of the DYMGNN can be reproduced under strict memory constraints. By implementing stochastic sampling and a node-independent attention mechanism, the replication achieved an AUC of 0.8854. However, the analysis revealed that explicit, metadata-based topologies are structurally fragile when stripped of temporal dynamics.

Second, the study proposes an Implicit Graph Paradigm (Hypothesis 2), which replaces static metadata connections with a dynamic "Implicit Topology" constructed via behavioral k-Nearest Neighbors. This approach hypothesizes that risk propagates more effectively through behavioral homophily than geographic co-location. Empirical results confirm this: the implicit model achieved a statistically significant improvement in classification accuracy (+5.22 pp) and F1 score relative to the explicit baseline, while increasing the AUC to 0.8895. Complementary SHAP and reliability analyses confirm that the architecture captures non-linear risk dynamics, such as delinquency regime switches, with stable calibration. These results indicate that implicit, feature-driven graph construction represents a promising approach for developing robust and scalable credit risk models.

Keywords: Graph Neural Networks, Credit Risk, Dynamic Graphs, Implicit Topology, DYMGNN, Scalability

Zusammenfassung

Graph Neural Networks (GNNs) stellen den neuesten Stand der Technik in der Kreditrisikomodellierung dar und integrieren die Konnektivität der Kreditnehmer mit zeitlichen Dynamiken. Bestehende Frameworks wie das von Zandi et al. (2025) vorgeschlagene DYMGNN basieren jedoch auf rechenintensiven Vollclique-Topologien und starren Metadaten, was ihre Einsatzfähigkeit in ressourcenbeschränkten Umgebungen einschränkt. Diese Arbeit befasst sich mit diesen Einschränkungen, indem sie skalierbare, verhaltensgesteuerte topologische Alternativen auf hochwertiger Verbraucherhardware untersucht.

Die Studie verläuft in zwei methodischen Phasen. Zunächst wird mit einer Constraint-Aware Replication (Hypothese 1) bewertet, ob die Vorhersagelogik des DYMGNN unter strengen Speicherbeschränkungen reproduziert werden kann. Durch die Implementierung von stochastischem Sampling und einem knotenunabhängigen Aufmerksamkeitsmechanismus erreichte die Replikation einen AUC-Wert von 0,8854. Die Analyse ergab jedoch, dass explizite, metadatenbasierte Topologien strukturell fragil sind, wenn sie ihrer zeitlichen Dynamik beraubt werden.

Zweitens schlägt die Studie ein implizites Graph-Paradigma (Hypothese 2) vor, das statische Metadatenverbindungen durch eine dynamische „implizite Topologie" ersetzt, die über verhaltensbasierte k-Nearest Neighbors konstruiert wird. Dieser Ansatz geht von der Hypothese aus, dass sich Risiken durch Verhaltenshomophilie effektiver ausbreiten als durch geografische Ko-Lokalisierung. Empirische Ergebnisse bestätigen dies: Das implizite Modell erzielte eine statistisch signifikante Verbesserung der Klassifizierungsgenauigkeit (+5,22 pp) und des F1-Scores im Vergleich zur expliziten Basislinie, während der AUC-Wert auf 0,8895 stieg. Ergänzende SHAP- und Zuverlässigkeitsanalysen bestätigen, dass die Architektur nichtlineare Risikodynamiken, wie z. B. Wechsel des Delinquenzregimes, mit stabiler Kalibrierung erfasst. Diese Ergebnisse deuten darauf hin, dass die implizite, merkmalsbasierte Graphkonstruktion einen vielversprechenden Ansatz für eine wissenschaftlich robuste und operativ skalierbare Kreditrisikomodellierung darstellt.

Schlüsselwörter: Graph Neural Networks, Kreditrisiko, Dynamische Graphen

Acknowledgements

I would like to express my sincere gratitude to my supervisor, Dr. Rick Steinert, for his timely and insightful guidance throughout the development of this thesis. His expertise and support have been invaluable in shaping both the direction and quality of my research.

I am also grateful to the Finance and Capital Market Theory department for generously allowing me to use their powerful computing resources to train my models, which greatly facilitated the computational aspects of this work.

My thanks also go to my colleague, Andres Cadena (MSc IBA – DSDS student), for taking the time to review my thesis and provide constructive feedback.

Finally, I would like to acknowledge the University Writing Centre for their informative writing sessions, which helped me improve the clarity and presentation of my work.

In addition, I would like to thank Mr. Sahab Zandi, the key author of the reference model, for kindly taking the time to discuss the significant memory and scalability challenges inherent in the original framework. His remarks served as an early warning and prompted a critical reassessment of the replication strategy, directly influencing the methodological direction adopted in this thesis.

Table of contents

1. Introduction

1.1 State of the Art in Credit Risk Modeling

Credit default prediction is established as a foundational problem in financial risk analytics. Early parametric approaches, most notably Altman's Z-Score, demonstrated that multivariate discriminant analysis could significantly outperform simple univariate ratio heuristics, marking the transition toward data-driven default risk modeling (Altman 1968). Over the following decades, tree-based algorithms, support vector machines, k-nearest neighbors, and ensemble methods achieved higher predictive accuracy compared to traditional statistical models such as logistic regression or linear discriminant analysis (Barboza, Kimura, and Altman 2017). Despite these improvements, most models continued to treat borrowers as independent entities, which limited their ability to capture the relational and interconnected nature of financial behavior (Agarwal and Taffler 2007).

After the 2008 financial crisis, interest in network-based approaches increased significantly (Caccioli, Barucca, and Kobayashi 2018). Studies such as Óskarsdóttir and Bravo (2021) introduced multilayer network representations of borrower and lender relationships and applied algorithms like Personalized Multilayer PageRank. Their results showed that relational dependencies can contribute additional predictive power beyond models based solely on Independent and Identically Distributed (IID) assumptions.

However, these initial network analytics approaches have certain methodological limitations: they treat topological information (structure) and attribute information (finance data) as separate entities (J. Wang et al. 2022). Metrics like PageRank are pre-computed as static features and then fed into standard classifiers, meaning the model cannot learn how the graph structure specifically influences the risk of a specific borrower during the training process. Consequently, the topology remains an extrinsic input rather than an intrinsic part of the learning mechanism.

To resolve this separation, Graph Neural Networks (GNNs) have emerged as the state-of-the-art paradigm. By employing a differentiable neural message-passing mechanism, GNNs fuse node attributes with network topology into a unified learned representation (Hamilton 2020). Within this domain, recent literature has proposed specialized architectures to address specific financial complexities:

- **Higher-Order Structures:** MotifGNN (D. Wang et al. 2020) extends standard message passing to capture repeated connection patterns among multiple nodes (called motifs) which reflect complex interactions among groups of borrowers rather than just pairwise relationships. It uses curriculum learning to focus on rare subgraphs, which are especially useful for detecting unusual or risky behavior that simple edge-by-edge analysis might miss.
- **Data Heterogeneity:** Addressing the diversity of financial data, Multi-view Graph Convolutional Networks construct multiple distinct relational graphs (e.g., separate views for transaction history, social ties, and guarantor links). These views are synthesized using a gating mechanism, allowing the model to leverage complementary relational patterns and significantly improve robustness compared to single-view architectures (Z. Li et al. 2024).

Beyond static structural analysis, the integration of temporal dynamics has emerged as a critical frontier in credit risk modeling. Addressing the time-varying nature of default risk, architectures such as TemGNN (D. Wang et al. 2021) have successfully synthesized static graph embeddings with short-term relational dynamics. By coupling graph convolutions with Long Short-Term Memory

(LSTM) modules, this approach explicitly models the evolution of borrower behavior rather than relying on a single historical snapshot.

Parallel to these temporal advancements, attention-based mechanisms have refined how models interpret heterogeneous graph structures. For instance, Zhou et al. (2023) utilized Graph Attention Networks (GAT) to process multiple borrower similarity graphs, enabling the model to dynamically learn which relational signals are most informative. Their findings confirm that distinguishing the relevance of specific neighbors via attention weights, rather than treating all connections equally, yields superior predictive performance compared to architectures restricted to isolated borrower features or unweighted graphs.

Overall, the research trend in credit-risk modeling has shifted from isolated borrower-level prediction toward methods that represent borrowers within financial, behavioral, and temporal networks. Recent studies increasingly emphasize that relational structure is not simply an auxiliary signal but a core component of predictive performance. It is at this specific intersection of structural adaptivity and temporal modeling that the "Attention-based dynamic multilayer graph neural networks for loan default prediction" (DYMGNN) by Zandi et al. (2025) is situated (henceforth referred to as the " "Reference Model"). By integrating the neighbor-ranking capability of GAT with the sequential memory of LSTMs, the DYMGNN framework aims to capture the holistic "lifecycle" of credit risk.

1.2 Motivation and Problem Definition

While the DYMGNN framework proposed by Zandi et al. (2025) represents a significant advancement in capturing the complex, time-varying dynamics of credit risk, its deployment in certain operational environments presents specific engineering challenges.

1. **The Scalability Challenge:** To achieve its high predictive accuracy, the Reference Model leverages a sophisticated architecture combining GAT with LSTM layers. By design, this approach requires constructing comprehensive graphs based on explicit metadata (e.g., full connectivity between all borrowers of a specific lender). While theoretically optimal, this structure results in quadratic memory complexity ($O(N^2)$). Topology analysis conducted in this study quantifies the magnitude of this bottleneck: a direct replication on a standard dataset generates over 10 billion edges, resulting in a theoretical memory requirement of 3.78 TB VRAM, exceeding the capacity of high-end consumer hardware (24 GB) by a factor of over 150. A key motivation of this study is, therefore, to investigate "Constraint-Aware" adaptations, determining how this state-of-the-art architecture can be optimized to function efficiently on top-tear consumer-grade GPU without significant performance degradation.
2. **Generalizing Beyond Explicit Metadata:** Generalizing Beyond Explicit Metadata: Furthermore, the Reference Model defines network edges exclusively through explicit shared attributes (e.g., Lender, Zip Code), confining dynamic behavioral signals (e.g., repayment dynamics) to individual node features. While this "Explicit Topology" provides a robust signal, recent literature suggests that using latent behavioral patterns to define structural connectivity might offer a complementary source of risk information (Zhou et al. 2023).

Expanding the DYMGNN framework to incorporate "Implicit Topologies" (learned dynamically via feature similarity) serves two critical objectives: it significantly enhances predictive power by allowing risk propagation based on behavioral affinity rather than just static location, and it

improves operational efficiency by eliminating the computational bottleneck of maintaining complex external metadata tables.

Research Gap: Consequently, this study aims to bridge the gap between theoretical performance and operational flexibility. It seeks to validate whether the powerful logic of the DYMGNN can be:

1. Replicated under strict memory constraints (Scalability).
2. Extended to derive network structures directly from borrower behavior (Implicit Topology)

1.3 Research Objectives and Hypotheses

This study addresses the computational and practical constraints of the DYMGNN framework through two primary objectives:

H1 (Constraint-Aware Replication): The Reference Model's predictive logic can be reproduced on consumer-grade hardware through topological approximation.

The primary objective is to evaluate whether the DYMGNN architecture, originally designed for high-performance computing clusters, can be effectively replicated on constrained hardware (e.g., single GPU) without structural failure or predictive collapse. This hypothesis tests the operational feasibility of the model, quantifying the trade-offs between strict hardware limitations and the integrity of the risk rankings.

H2 (Proposed Model): An implicit, feature-driven graph structure (Implicit Topology) serves as a viable substitute for explicit metadata graphs, achieving comparable predictive performance while resolving quadratic memory bottlenecks.

This objective evaluates whether an implicit graph topology constructed using k-Nearest Neighbors (KNN) can serve as an effective alternative to costly, metadata-based explicit graphs. Unlike explicit graphs that depend on rigid metadata (e.g., Lender), KNN graphs are built from behavioral feature similarities. The hypothesis is that a feature-driven implicit topology can better capture latent risk correlations, offer greater scalability, and maintain predictive accuracy.

1.4 Scope and Limitations

Primary Objective. This study focuses on the prediction of one-year-ahead loan default using high-dimensional behavioral data. The target variable is defined as the first occurrence of serious delinquency (90 days past due) within a 12-month prediction horizon.

Data and Geographic Scope. The analysis is strictly limited to the US Single-Family Mortgage Market, utilizing the Freddie Mac Single-Family Loan-Level Dataset.

Network Definition. The explicit topological analysis is restricted to two specific dimensions of connectivity: Geographic Co-location (2-digit Zip Code) and Institutional Linkage (Mortgage Lender). Other potential social or transactional connections are outside the scope of the explicit baseline.

Single-Source Limitation. As the data originates solely from a Government-Sponsored Enterprise (GSE), the findings reflect the risk profile of "conforming loans" and may not generalize to the subprime market or private-label securitizations.

Origination. The analysis is restricted to loans originated in 2009 and 2010, a period heavily influenced by the aftermath of the Global Financial Crisis. This introduces a "Crisis Bias," where learned patterns may reflect stressed economic conditions rather than stable market dynamics.

Observation Window. The model training utilizes 18 months of behavioral performance data (Jan 2012 – June 2013), with out-of-sample testing conducted on the subsequent 6 months (July 2013 – Dec 2013).

- **Optimization of Sequence Length** While the Reference Model suggests that extending the lookback period is possible, this study strictly enforces a fixed window of $\tau = 6$ based on two methodological reasoning:
  - **Signal-to-Noise Ratio:** As illustrated in the attention analysis of the reference study (Zandi et al., 2025), the normalized attention scores $\beta^{(t)}$ exhibit a non-linear, exponential decay as the time lag increases. The model allocates the majority of its importance mass to the final three snapshots ($t \in \{4, 5, 6\}$), confirming that the "Effective Information Horizon" for default prediction is short-term. Extending the sequence beyond six months ($\tau > 6$) would introduce structural noise, feeding the LSTM non-informative timesteps that dilute the gradient signal during backpropagation without adding recoverable predictive value.
  - **Hardware-Imposed Bounds:** Extending the sequence length imposes a linear increase in the memory footprint required for Backpropagation Through Time (BPTT). Since the GNN encoder must generate embeddings for every snapshot in the sequence, a window of $\tau = 12$ would effectively double the VRAM requirement for storing intermediate activation maps compared to $\tau = 6$. Given that this study operates near the 24GB VRAM limit of the consumer-grade hardware to maintain graph density, increasing τ would necessitate further aggressive neighbor truncation to avoid Out-Of-Memory (OOM) errors. We prioritize maintaining topological fidelity (higher neighbor count) over an extended temporal window that offers diminishing predictive returns.

Hardware-Imposed Boundaries. Hardware-Imposed Boundaries (Constraint-Awareness). This study is conducted utilizing high-end consumer-grade hardware. Despite this computational capacity, the full-clique topology of the Reference Model is approximated using a Hybrid Sampling Strategy and aggressive neighbor truncation (e.g., capping neighborhoods) to satisfy memory limitations. The results therefore represent the performance of the architecture under representative resource constraints, distinct from its theoretical maximum capacity on industrial-scale computing clusters.

2. Theoretical Background

2.1 The Graph Neural Networks (GNN)

GNNs are specialized deep learning architectures designed to operate on irregular, non-Euclidean data structures where standard operations like Convolution (CNNs) or recurrence (RNNs) are not

directly applicable.

The fundamental intuition behind GNNs is the Homophily Assumption: nodes that are connected in a graph are likely to share similar properties or labels. To leverage this, GNNs employ a Neural Message Passing framework. As described by Hamilton (2020) and illustrated in Figure 1, this framework consists of an Encoder, which maps nodes to low-dimensional latent embeddings, and a Decoder, which utilizes these embeddings to perform specific downstream prediction tasks.

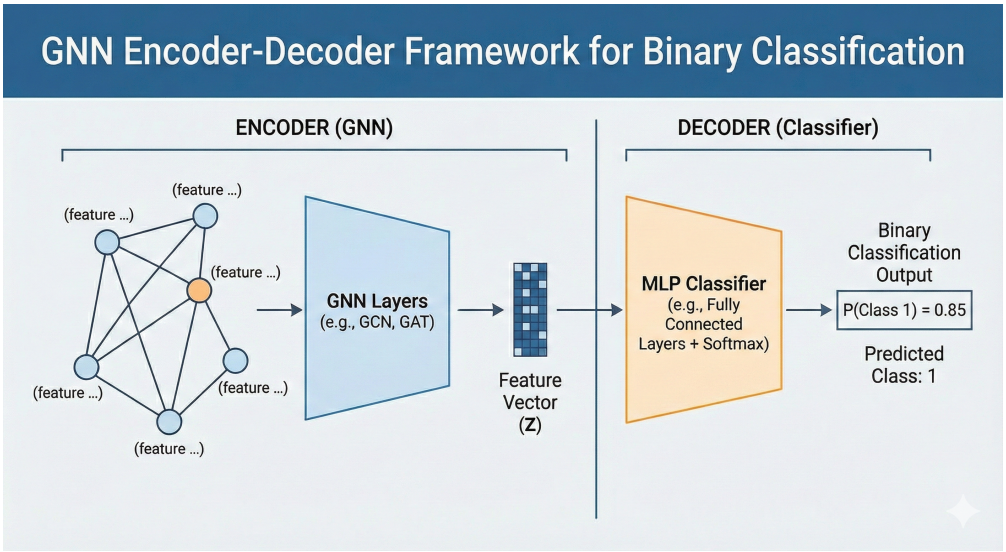


Figure 1: GNN Encoder-Decoder Framework. AI-generated by the author.

2.1.1 The Encoder: Message Passing Mechanism

The core of the GNN encoder is the iterative propagation of information between connected nodes. This process allows a node to aggregate feature information from its local neighborhood, effectively updating its own representation based on the context of its peers.

A general GNN layer can be expressed, according to Hamilton (2020), using the abstract **UPDATE** and **AGGREGATE** functions as follows:

$$h_u^{(k+1)} = \text{UPDATE}^{(k)}\left(h_u^{(k)}, \text{AGGREGATE}^{(k)}(\{h_v^{(k)} \mid v \in \mathcal{N}(u)\})\right) \tag{1}$$

where $h_u^{(k)}$ is the representation of node u at layer k, $\mathcal{N}(u)$ denotes the neighbors of node u, $\text{AGGREGATE}^{(k)}$ combines information from neighbors, and $\text{UPDATE}^{(k)}$ applies a nonlinear transformation to produce the updated node embedding (Figure 2).

Crucially, stacking K such layers allows the model to capture structural information from K-hops away. For example, a 2-hop GNN allows a borrower node to be influenced not just by their direct co-borrowers, but by the "neighbors of neighbors."

A canonical implementation of this framework (often referred to as a Graph Convolutional operation) utilizes a linear transformation followed by a non-linear activation, as formulated below:

$$h_u^{(k+1)} = \sigma\Big(W_{\text{self}}^{(k)} h_u^{(k)} + W_{\text{neigh}}^{(k)} \sum_{v \in \mathcal{N}(u)} h_v^{(k)} + b^{(k)}\Big) \tag{2}$$

In this formulation:

- $W_{\text{self}}^{(k)}$ and $W_{\text{neigh}}^{(k)}$ are learnable weight matrices for the node itself and its neighbors, respectively.
- $b^{(k)}$ is a bias term, and σ is a nonlinear activation function (e.g., ReLU or sigmoid).
- The summation over neighbors replaces the abstract AGGREGATE function, and the linear transformation with activation implements the UPDATE function.

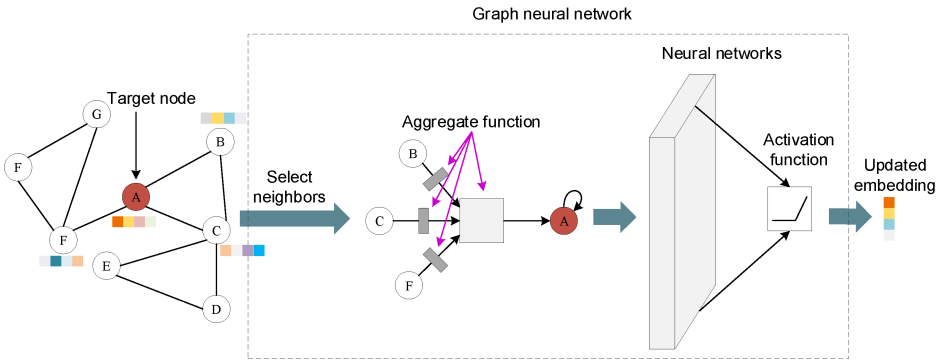


Figure 2: The Encoder Framework. Adapted from Zeng & Tang (2021).

2.1.2 The Decoder

The decoder acts as the prediction module that transforms learned node or graph embeddings into task-specific outputs, such as node classifications, edge probabilities, or reconstructed graph structures. It typically applies a scoring or similarity function (e.g., Multi-Layer Perceptrons) to interpret the latent representations produced by the GNN encoder.

2.2 The Reference Model: DYMGNN

This section details the specific architecture of the Reference Model. The DYMGNN is a hybrid architecture that sequentially processes structural information (via GAT) and temporal evolution (via LSTM).

2.2.1 The Topological Layer: Graph Attention Networks (GAT)

The first component of the Reference Model addresses the spatial dependencies between borrowers. As described in the Reference Model, while the input graph constructed for this study is unweighted (binary edges), treating all neighbor connections equally, as done in standard isotropic Graph Convolutional Networks (GCN), is suboptimal for risk modeling. In financial networks, the influence of neighbors is heterogeneous; a connection to a defaulted borrower should theoretically carry more weight than a connection to a standard borrower.

To capture this nuance, the GAT introduces a mechanism to learn the strength of relationships dynamically. Unlike GCNs which use fixed normalization, GAT computes a learnable attention coefficient α_{ij} for every edge, representing the "importance" of neighbor j's features to node i.

The attention mechanism computes a pairwise importance score between a central node i and its neighbor j as follows:

$$e_{ij} = \text{LeakyReLU}\big(\mathbf{a}^\top \left[W\mathbf{X}_i \parallel W\mathbf{X}_j\right]\big) \tag{3}$$

Here, $\parallel$ denotes the concatenation operation, and $\mathbf{a}^\top$ is a learnable weight vector. These raw scores are then normalized using a softmax function to make them comparable across different neighbors:

$$\alpha_{ij} = \frac{\exp(e_{ij})}{\sum_{k \in \mathcal{N}(v_i) \cup \{v_i\}} \exp(e_{ik})} \tag{4}$$

Finally, the node embeddings are updated by aggregating the neighbor features scaled by these normalized attention scores:

$$\mathbf{Z}_i = \sum_{j \in \mathcal{N}(v_i) \cup \{v_i\}} \alpha_{ij} W\mathbf{X}_j \tag{5}$$

To stabilize the learning process, GAT architectures often employ Multi-Head Attention, where K independent attention mechanisms execute the transformation of Equation (5) in parallel. The Reference Model's exact head configuration was implicit.

2.2.2 The Temporal Layer: LSTM

While the GAT layer captures the spatial relationships at a specific snapshot in time, borrower risk is inherently dynamic. A single missed payment might be an anomaly or the start of a trend. To distinguish between these scenarios, the LSTM network is employed to model the sequential evolution of the borrower's state.

Designed to overcome the "vanishing gradient" problem of standard Recurrent Neural Networks (RNNs), LSTMs utilize a sophisticated gating mechanism to regulate the flow of information over long sequences (Goodfellow, Bengio, and Courville 2016). This allows the model to retain critical historical context while adapting to recent behavioral changes.

The LSTM maintains a cell state $C^{(t)}$ and hidden state $H^{(t)}$ updated via the following gates:

$$\begin{aligned}
I^{(t)} &= \sigma(W_{X,H,2} \cdot [H_{t-1}, X_t] + b_2) \\
F^{(t)} &= \sigma(W_{X,H,1} \cdot [H_{t-1}, X_t] + b_1) \\
\tilde{C}^{(t)} &= \tanh(W_{X,H,3} \cdot [H_{t-1}, X_t] + b_3) \\
O^{(t)} &= \sigma(W_{X,H,4} \cdot [H_{t-1}, X_t] + b_4) \\
C^{(t)} &= F^{(t)} \odot C^{(t-1)} + I^{(t)} \odot \tilde{C}^{(t)} \\
H^{(t)} &= O^{(t)} \odot \tanh(C^{(t)})
\end{aligned} \tag{6}$$

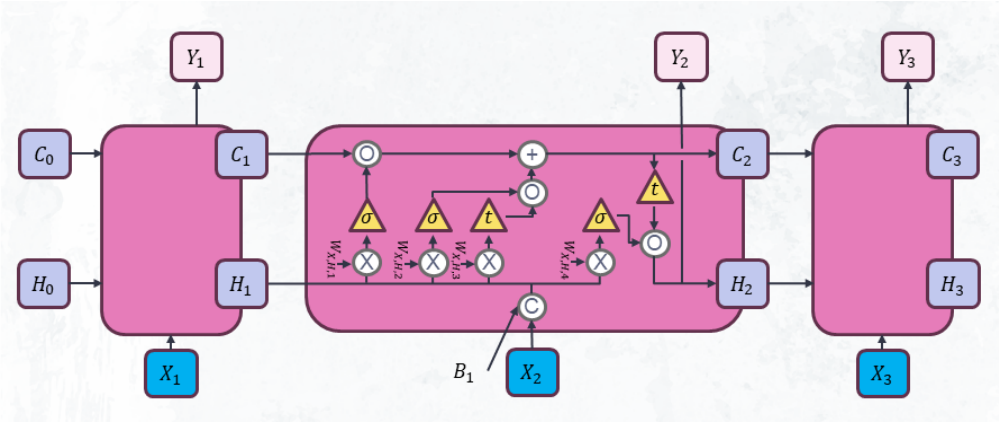


Figure 3: LSTM Diagram. Adapted from Steinert (2025)

Input Gate ($I^{(t)}$). Uses $W_{X,H,2}$, controlling the flow of new candidates.

Forget Gate ($F^{(t)}$). Uses $W_{X,H,1}$, controlling the retention of the previous cell state.

Output Gate ($O^{(t)}$). Uses $W_{X,H,4}$, controlling the final hidden state.

By stacking the LSTM on top of the GAT, the Reference Model effectively learns a spatio-temporal embedding, unifying the influence of a borrower's neighbors with the momentum of their own repayment history.

2.2.3 Temporal Attention

While the LSTM captures sequential dependencies, not all time steps are equally indicative of default risk. A borrower may exhibit stable behavior for months, followed by a sudden "shock" event. Standard recurrent models, which often rely on the final hidden state or simple mean pooling, may suffer from information bottlenecking or signal dilution.

To capture the "Global Variation of Trends," the Reference Model employs a soft attention mechanism applied to the sequence of hidden states output by the LSTM. Unlike simple mean pooling, which treats all time steps as equally informative, this global attention-based pooling mechanism learns to assign differential importance to each historical snapshot, allowing the model to focus on specific periods of financial stress or behavioral change.

Let $H^{(t)}$ denote the hidden states at different timestamps.The attention score for each hidden state is first computed as:

$$s^{(t)} = a_h H^{(t)} W_h \tag{7}$$

where $a_h \in \mathbb{R}^{1\times n_l}$ and $W_h \in \mathbb{R}^{D\times 1}$ are learnable weight vectors. These scores are then normalized across all timestamps using a softmax function:

$$\beta^{(t)} = \frac{\exp(s^{(t)})}{\sum_{k=1}^{\tau} \exp(s^{(k)})} \tag{8}$$

The final aggregated hidden state is obtained by weighting each timestamp by its normalized attention score:

$$\hat{H}_{att} = \sum_{t=1}^{\tau} \beta^{(t)} H^{(t)} \tag{9}$$

This process effectively re-weights the contribution of each timestamp, allowing the model to focus on the most relevant temporal snapshots.

2.2.4 Decoder

The decoder employed in the Reference Model is a feed-forward neural network designed to map the learned node embeddings to a final default probability. Its architecture consists of a dense layer reducing the input dimension from 20 to 10 with ReLU activation and dropout regularization ($p = 0.5$), followed by a final dense layer with a sigmoid activation function to generate the binary classification output.

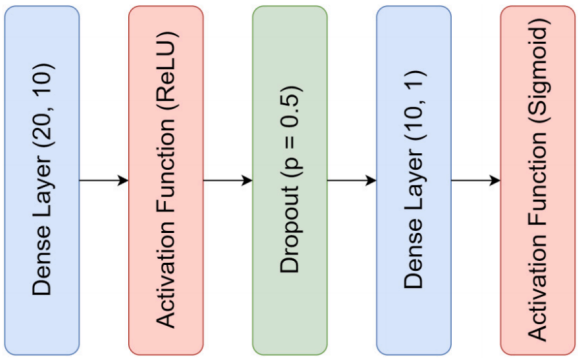


Figure 4: Architecture of the Decoder in the Paper

The model is optimized using Binary Cross-Entropy (BCE) loss. It measures the difference between the true labels and the predicted probabilities by penalizing predictions that deviate from the ground truth.

$$\text{Loss} = -\frac{1}{n} \sum_{i=1}^{n} \left[Y_i \cdot \log(\hat{Y}_i) + (1 - Y_i) \cdot \log(1 - \hat{Y}_i) \right] \tag{10}$$

where n is the number of samples, Y_i is the true label for sample i, and $\hat{Y}_i$ is the predicted probability for sample i.

2.3 Model and Experimental Setup

To evaluate the proposed architecture, Zandi et al. (2025) established a specific experimental setup involving rolling window validation and dynamic graph construction. This framework, which this study aims to replicate, is defined as follows:

Network Structure and Data. The network consists of two layers defined by geographical location (Zip code) and lender (mortgage provider). The training set comprises 148,520 nodes for training and 96,490 nodes for test for single-layer networks, making it totally ~250,000 unique loans.

Features. The model utilizes total of 16 feature consisting of both static application (e.g., FICO, DTI) and dynamic behavioral (e.g., current unpaid balance, months remaining) features.

Node Dynamics. To retain neighborhood information, a loan that defaults remains in the graph as a defaulter for the duration of the current observation window. It is removed from the graph only once the rolling window moves past the default event.

Rolling Window Dynamics. While the topology is fixed within a window, the graph evolves between windows. As the rolling window shifts forward by one month, loans that have concluded or defaulted (outside the observation logic) are removed, and the set of active nodes (V) is updated, resulting in a new static supra-graph for the subsequent window.

Training Configuration: The model uses 18 months of behavioral data (Jan 2012 – June 2013) for training and a subsequent 6 months (July 2013 – Dec 2013) for testing.

2.4 Benchmark Results of the Reference Model

Performance Gains. Overall, the results reported by Zandi et al. demonstrate that incorporating dynamic network information provides a measurable advantage over static or feature-only approaches. This suggests that the model captures richer temporal and relational dependencies relevant for default-risk prediction. Relative to the strongest baseline (XGBoost), the DYMGNN architecture achieves gains of 0.87% in AUC and 1.67% in F1. While these improvements appear modest, they are meaningful in high-stakes financial settings where small increases in accuracy translate to significant capital savings.

Temporal Risk Propagation. A central finding of the reference study is that default risk is not an isolated event but a contagion process that propagates through the network over time. The authors explicitly emphasize this mechanism, noting that traditional models fail because they treat borrowers as independent entities, ignoring how "default risk propagates over a network". To address this, their dynamic approach ensures that "a loan that has defaulted will remain marked as a defaulter for the observation window". This design decision, effectively creating "ghost nodes" that persist after default, allows the model to capture how past financial distress in a neighborhood or lender portfolio "infects" new applications, a feature the authors identify as the core driver of their performance gains.

The "Static Topology" Gap. Crucially, the reference results highlight a significant limitation in static graph approaches. The Static GAT baseline (AUC 0.763) underperformed the non-graph XGBoost baseline (AUC 0.805) by a wide margin. This suggests that the explicit lender-based topology, when stripped of temporal dynamics, introduces more noise than signal. This observation reinforces the motivation for H2, which seeks to determine if a feature-driven implicit topology (KNN) can recover this lost signal more effectively than the rigid metadata graphs used by Zandi et al.

Model Stability. The dynamic model yields narrower confidence intervals compared to both non-GNN baselines (XGBoost, Logistic Regression, DNN) and GNN-based baselines (GAT, GCN).

This indicates higher stability and robustness, likely due to the model's ability to leverage network effects to stabilize predictions for individual borrowers.

Metric Ambiguity. A critical omission in the reference study is the lack of specification regarding F1 score computation. It remains unclear whether the reported F1 of 0.851 utilizes a standard threshold ($p = 0.5$), an optimized threshold (maximized on validation data), or weighted averaging. Given the severe class imbalance inherent in default prediction, the choice of threshold drastically alters the metric. Consequently, exact replication of the F1 score may prove elusive without this hyperparameter definition.

Table 1: Best Results Reported by Zandi et al. (2025)

Model	AUC (mean ± CI)	F1 (mean ± CI)
DYMGNN	0.812 ± 0.008	0.851 ± 0.007
XGBoost	0.805 ± 0.018	0.837 ± 0.012
Static GAT	0.763 ± 0.014	0.817 ± 0.012

3. Methodology and Implementation Strategy

3.1 Methodological Framework and Implementation Adaptations

3.1.1 Consistent Across Hypotheses

To ensure a rigorous comparison between the Reference Model and the proposed hypotheses, this study establishes a common data processing baseline. The target definition and cleaning procedures described below apply uniformly to both the H1 (Replication) and H2 (Proposed) workflows.

Sampling Strategy. Following the data preparation stage, the dataset comprised approximately 2.7 million unique loans. To ensure a rigorous methodological comparison with the Reference Model (which utilized a total of ~250,000 unique loans) and to satisfy computational constraints, we did not utilize the full 2.7 million records. Instead, the following sampling strategy was employed to mimic the exact scale and network structure of the Reference Model dataset:

- **Stratified Sampling:** A subset of 250,000 loans was sampled from the cleansed dataset using stratification across geographic regions and lenders to preserve the original network structure.
- **Class Balance Consideration:** The Reference Model reports a default rate of approximately 5%. To mirror this, the sampling procedure targeted a 5% default ratio. However, due to the rolling-window construction, where defaulted loans exit the observation window over time, the effective default rate within the active training windows naturally varied between 2% and 3%. This accurately reflects the temporal drift inherent in real-world credit portfolios.

Data Preparation. In addition to the preprocessing steps described in the Reference Model, supplementary data filtering procedures were applied to ensure dataset integrity.

- **Default Timer Filter:** A "Default Timer" logic was introduced to remove loans that had already reached a terminal default state prior to the start of the training window (Jan 2012). Calculating the months elapsed since the dataset reference point (February 2009), any loan defaulting before month 35 was excluded. This ensures the model is trained exclusively on active, "at-risk" loans, rather than learning trivial patterns from historical defaults.
- **Window Filter:** Loans outside the target analysis period (2009–2014) were removed to maintain consistency with the original temporal window and to keep managable dataset.
- **Freddie Mac Validity Filter:** Following Freddie Mac's published data-quality rules, observations with values outside the allowable ranges for specified features were excluded to ensure all retained records reflected valid and reliable loan information.

Target Variable Definition. The target variable is defined as the first occurrence of serious delinquency (90 days past due) within a 12-month prediction horizon following the observation window. To prevent data leakage:

- Masking: Loans that have already defaulted on or before the current snapshot are masked (assigned -1) to exclude them from the loss calculation.
- Labeling: Remaining active loans are labeled as positive (1) if a default event occurs within the horizon and negative (0) otherwise.

Handling of Foreclosures (RA). In the Freddie Mac dataset, the delinquency code 'RA' (REO Acquisition) indicates that a property has undergone foreclosure and been repossessed by the lender. Although not explicitly detailed in the original paper, this status represents a terminal credit loss (Freddie Mac 2024). Therefore, this study explicitly maps 'RA' to a delinquency severity of 3 or higher, ensuring these events are correctly classified as defaults.

Temporal Windowing and Burn-in Recovery. To strictly define the operational cadence, the experimental protocol utilizes 18 distinct rolling training windows spanning from January 2012 to June 2013. A direct replication of the Reference Model's timeline implies a 13 valid training windows, however the initial months (Jan–May 2012) are typically consumed for LSTM state initialization ("burn-in") when historical data is unavailable.

In contrast, this study maximizes data utilization by leveraging the dataset's extended origination history. By utilizing this "pre-history" (starting June 2011) to populate the 6-month Lookback Sequence ($t = 6$) for the earliest target months, the full 18 training windows were recovered without data censoring. This deviation enhances the robustness of the evaluation by extending the effective training distribution to cover a wider variety of economic sub-conditions than the constrained reference protocol.

Consistent to the Reference Model.

- **Imputation and Capping:** Median imputation (for numerical variables), mode imputation (for categorical variables), and 1st/99th percentile capping were applied using only the training data to prevent information leakage.
- **Scaling:** All numerical features were scaled using min–max normalization based on statistics derived from the training set, ensuring that features contribute proportionally during model optimization.

3.1.2 Hypothesis-Specific Deviations from the Reference Model

While H1 aims to replicate the Reference Model, certain architectural and training modifications were necessary to accommodate hardware constraints and to resolve numerical instabilities identified in the original formulation.

The modifications fall into three categories: (1) Scalability adaptations to manage memory constraints (graph topology, neighbor selection, implementation strategy), (2) Architectural refinements to address numerical stability issues (attention mechanism, GNN architecture, loss function), and (3) Experimental enhancements to maximize data utilization (training window extension, attention head specification).

H1 focuses primarily on constraint-aware replication using stochastic sampling of explicit metadata graphs, while H2 introduces a fundamentally different paradigm by constructing implicit, behavior-driven topologies with additional architectural stabilization mechanisms. Table 2 summarizes these strategic deviations, which will be explored in detail in the subsequent sections.

Table 2: Methodological Specifications and Necessary Deviations

Component	Reference Spec	H1 (Explicit)	H2 (Implicit)
Graph Topology	Full Clique	Stochastic (Random Sampling)	Adaptive KNN
Neighbor Selection	Connect All	Random Truncation (Cap at 50)	Adaptive Top-K ($K \propto \ln N$)
Attention Scope	Global Pooling	Node-Independent	Node-Independent
GNN Architecture	Standard GAT	Standard GAT	Stabilized GAT (Residuals + LayerNorm)
Loss Function	Standard BCE	BCE + Pos. Weight	Focal Loss + Pos. Weight
Implementation	Sequential Snapshots	Disjoint Union	Sequential Snapshots
Training Window	13 Windows	18 Windows	18 Windows
GAT Attention Heads	Unspecified	2	2

3.2 Hypothesis 1: Constraint-Aware Replication

3.2.1 Motivation: Constraint-Aware Replication

In this study, Hypothesis 1 is formally defined as a Constraint-Aware Replication. While the Reference Model demonstrated efficacy using high-performance computing (HPC) infrastructure, likely leveraging full-batch processing on industrial-grade clusters, this study investigates the operational feasibility of the architecture on a high-end consumer-grade hardware.

A direct 1-to-1 replication of the reference implementation reveals a critical scalability bottleneck: the construction of full clique graphs based on metadata results in quadratic memory complexity ($O(N^2)$). A topology analysis[1] of the stratified sampled dataset revealed the true magnitude of

[1]The script `topology_analysis.py` performs diagnostics of stochastic block sampling and calculates borrower-level neighbor statistics to recommend optimal maximum degrees for each layer.

this bottleneck. Without intervention, the native graph structure (based on 2-digit Zip codes and Lenders) generates an average node degree of 40,572, resulting in approximately 10.1 billion edges. This configuration entails a theoretical memory requirement of 3.78 TB of VRAM, exceeding the 24 GB capacity of the available hardware by a factor of over 150x.

To address this and ensure a scientifically rigorous reproduction of the model's logic without requiring its infrastructure, the following controlled adaptations were introduced:

- **Hybrid Connectivity Strategy:** Instead of constructing full cliques, a hybrid graph-building strategy was applied. Small borrower groups retained full connectivity, while larger groups were truncated using stochastic neighbor sampling (capped at 100 neighbors for geographic links and 200 for lender links). These limits were not arbitrary; they were imposed as strict hardware boundaries to reduce the graph from its unconstrained size (3.78 TB) to a manageable footprint.

  Note: This stochastic neighbor selection introduces an implicit form of structural regularization. By exposing the model to different subsets of neighbors across epochs, it mitigates overfitting to static graph structures, functionally replacing the explicit "50% node isolation" mechanism described in the reference model.

- **Static Supra-Graph with Masking:** Rather than reconstructing a new graph snapshot at each timestamp (which incurs prohibitive I/O overhead), a Static Supra-Graph (Disjoint Union) approach was utilized. All nodes and edges across the full time horizon are pre-loaded into a unified graph structure. During training, a dynamic "Temporal Mask" is applied to strictly zero out nodes that are inactive in a given window, substantially reducing memory overhead while preserving the temporal integrity of the LSTM sequences.

- **Mini-Batch Training:** Whereas the original study relied on full-batch training, this study employs Mini-Batch Processing. Subgraphs are sampled and processed sequentially, enabling scalable training on large dynamic graphs. This adaptation enables the DYMGNN framework to be democratized for non-HPC environments.

- **Final Graph Structure.** The final graph structure operates on a 'Supra-Adjacency' matrix of dimension $2N \times 2N$ (where $N = 250,000$ loans). This expansion allows the GNN to explicitly model auxiliary nodes (or temporal states) within the same message-passing step, resulting in a physical graph size of 500,000 nodes and approximately 148 million edges.[2]

Therefore, H1 serves not merely as a copy, but as a stress-test of the DYMGNN framework, determining if its predictive power holds under strict computational constraints.

3.2.2 Architectural Modifications

While the core GNN-LSTM framework remains faithful to the Reference Model, specific architectural components required modification to ensure numerical stability and convergence within the constraint-aware environment.

Multi-Head Attention. The Reference Model notes that employing multiple attention heads enhances representational capacity by allowing the model to attend to different neighbor sets simultaneously. However, the exact head count was unspecified. This study implements a two-head

[2]See `diagnostics_h1.py` for structural verification and exact node count validation.

attention mechanism ($K = 2$). This configuration provides a balance between capturing diverse topological features and maintaining the computational efficiency required for processing dense financial networks on consumer hardware.

Node-Independent Dynamic Attention. The Reference Model calculates temporal attention scores using fixed node indices (Equation 7). This introduces two limitations:

1. **Incompatibility with Dynamic Graphs:** The dependence on a fixed dimension N makes the model incapable of handling variable-sized graphs (where $N_t \neq N_{t+1}$), which is inherent to the rolling-window approach.
2. **Global Oversmoothing:** By projecting the entire graph state to a single global scalar, the mechanism forces all nodes to share the same temporal importance weights, ignoring individual borrower trajectories (Hamilton 2020).

To address both issues, the thesis implemented a Node-Independent Temporal Attention mechanism. Instead of pooling scores globally, our model computes attention weights for each node individually using a shared **Bahdanau-style MLP** (Bahdanau, Cho, and Bengio 2015):

$$e_i^{(t)} = W_2 \cdot \tanh\Big(W_1 h_i^{(t)}\Big) \tag{11}$$

where W_1 and W_2 are learnable weight matrices shared across all nodes. The tanh activation ensures numerical stability. These scores are normalized per node via softmax:

$$\alpha_i^{(t)} = \frac{\exp\Big(e_i^{(t)}\Big)}{\sum_{k=1}^{T} \exp\Big(e_i^{(k)}\Big)} \tag{12}$$

Finally, the context vector h_i^{att} is computed as the weighted sum of the node's temporal embeddings:

$$h_i^{\text{att}} = \sum_{t=1}^{T} \alpha_i^{(t)}\, h_i^{(t)} \tag{13}$$

This formulation is permutation-invariant[3] with respect to the node set but preserves node-level granularity. By allowing each borrower to have a distinct attention profile, we prevent the oversmoothing of risk signals while supporting variable-sized dynamic subgraphs (Hamilton 2020).

[3] A function is *permutation-invariant* if its output remains unchanged under any reordering of its inputs. In graph neural networks, this guarantees that node embeddings are independent of node or neighbor ordering, relying solely on graph structure and features.

3.2.3 Training Configuration

To stabilize convergence and manage the computational overhead of the DYMGNN architecture, the following specific training protocols were established.

Cost-Sensitive Loss. Due to a strong class imbalance (~19 non-defaults per default), we employ a Cost-Sensitive Binary Cross-Entropy (BCE) Loss. A scaling factor (`pos_weight`) amplifies the penalty on positive-class errors, prioritizing recall for high-risk loans while avoiding information loss from downsampling (Krawczyk 2016).

Hierarchical Neighbor Sampling. Processing high-degree nodes (which, as noted in the topological analysis, average node degree of 40,572) creates computational bottlenecks that exceed GPU memory limits. This study implements a Two-Level Hierarchical Sampling strategy to resolve this:

1. **Static Construction Caps:** These thresholds (100 Geo / 200 Lender) were empirically determined (testing on multiple thresholds) to ensure the static graph structure remained manageable in system RAM. Given the unconstrained high density neighbors, these caps serve as the primary filter to prevent immediate memory exhaustion during graph construction.
2. **Dynamic Stochastic Sampling:** To resolve the remaining memory bottleneck, the NeighborLoader was configured to dynamically sample a random subset of neighbors (set to $K = 50$) for each node in the active batch. Statistically, this threshold is grounded in the Central Limit Theorem, which posits that a sample size of $N \geq 30$ is typically sufficient to approximate the true mean of an underlying distribution. By selecting $K = 50$, we ensure that the aggregated neighborhood features serve as an unbiased and representative estimator of the full dense cluster, preserving the signal integrity despite the massive reduction in computational cost.

   Simultaneously, this technique acts as a form of Stochastic Edge Dropout, exposing the model to different local neighborhoods across epochs. As noted by Hamilton (2020), this not only stabilizes GPU memory usage but also serves as a structural regularizer, preventing overfitting to specific static connections. Methodologically, this mechanism serves as the constraint-aware alternative to the '50% random node isolation' employed by the Reference Model, achieving comparable structural regularization while ensuring memory feasibility.

Initialization and Verification. Prior to full-scale training, the GNN architecture underwent a 'Learnability Sanity Check.'[4] The model was trained on a single micro-batch of 32 nodes for 15 epochs. The loss decreased monotonically ($0.61 \rightarrow 0.13$), confirming that the custom message-passing layers and gradient propagation paths were correctly implemented and capable of learning non-linear patterns.

Replication Disclaimer. The results of Hypothesis 1 should be interpreted through the lens of "Constrained Replication." While the core modeling assumptions, loss objectives, and architectural logic of the Reference Model are strictly preserved, any observed performance deviations may partly reflect these necessary computational adaptations (e.g., neighbor truncation) rather than fundamental theoretical differences.

[4] See diagnostic_h1.py for the complete validation script.

3.3 Hypothesis 2: Proposed Model

3.3.1 Motivation: The Implicit Graph Paradigm

Replicating the Reference Model using the explicit graph approach (H1) confirmed the prohibitive computational overhead of metadata-based clique formation. While optimized sampling successfully reduced the memory cost from quadratic ($O(N^2)$) to linear ($O(N)$), it may have imposed a hard ceiling on topological fidelity. The necessary truncation of dense clusters diluted the risk signal, as evidenced by the performance collapse of the static baselines as further detailed in section 5.

To address these constraints, Hypothesis 2 proposes an Implicit Topology Framework utilizing KNN. In this paradigm, the graph structure is not strictly defined by external metadata (e.g., "Same Zip Code") but is learned dynamically from the data itself. By connecting each borrower to their k most similar peers in the behavioural feature space, the model posits that risk propagates most effectively among borrowers who exhibit similar financial trajectories, regardless of their physical proximity.

3.3.2 Feature Engineering for Similarity Measures

For a KNN algorithm to construct a semantically meaningful topology, the underlying distance metric must be robust. Raw magnitude features (e.g., current_upb in dollars) can distort distance calculations, causing the graph to cluster loans simply by size rather than risk behavior.

Therefore, H2 introduces additional refined features designed to normalize borrower behavior and track the true lifecycle of the loan. These features serve as the basis for both the topological construction (neighbor search) and the node learning:

- **Modification Flag (`is_modified`)**: Indicates whether the loan has been modified in the current or a prior period. While the dataset does not always specify the reason, this flag generally reflects adjustments made to prevent delinquency or foreclosure (Freddie Mac 2024).
- **Continuous Loan Age (`clean_loan_age`)**: The original Loan Age resets upon loan modification, obscuring the true lifecycle of the loan. This engineered feature tracks the borrower's cumulative progression, helping the model assess risk based on time spent in potentially vulnerable periods.
- **Unpaid Balance Percentage (`upb_pct_remaining`)**: Exploratory analysis (EDA) of the H2 dataset identified near-perfect multicollinearity ($\rho = 0.91$) between orig_upb and current_upb, confirming that the raw balance is a deterministic function of the loan's origin rather than its current risk state. Consequently, upb_pct_remaining was engineered to orthogonalize this signal, allowing the KNN algorithm to cluster borrowers based on repayment velocity rather than loan magnitude.

The incorporation of these engineered behavioral metrics increases the input feature space of H2 to 18 variables, distinct from the 16-variable schema utilized in the H1 replication and the original Reference Model (see Appendix A.1 for the full listing).

3.3.3 Adaptive Graph Construction Strategy

Edges between borrowers are constructed using a dynamic KNN strategy that adapts to cluster size and temporal changes:

- **Adaptive Neighbor Selection:** A base number of neighbors is defined (BASE_K_GEO = 6, BASE_K_LENDER = 7) based on the smallest group size and their square root divided by 2 which found to be the optimal size to start with given constraints. Small groups are fully connected to capture dense local interactions. For larger groups, the number of neighbors scales sublinearly using the heuristic $k \approx \max(\text{base}, \sqrt{n})$. This ensures that each borrower is connected to a relevant set of peers without creating computationally prohibitive dense blocks.
- **Neighbor Capping:** To enforce strict memory limits, the number of neighbors per borrower group is capped based on group size. Large groups employ a logarithmic cap to prevent overly dense neighborhoods. This guarantees that memory usage scales linearly $O(N)$ while maintaining model expressiveness.
- **Edge Dynamicity:** A critical innovation of H2 is the **behavioral evolution** of the graph structure. While both H1 and the Reference Model recalculate edges monthly as loans exit the portfolio, the connection logic differs fundamentally:
  - **H1 (Metadata-based):** Edges are determined by static attributes (Zip Code, Lender). If two borrowers share the same Zip Code, they remain connected as long as both loans are active, regardless of how their behaviors diverge.
  - **H2 (Behavior-based):** The network topology adapts to evolving risk profiles. By utilizing time-variant behavioral features (e.g., upb_pct_remaining, clean_loan_age) to compute nearest neighbors at each snapshot, the model captures the trajectory of risk:
    - Month : Borrower A is connected to Borrower B (both healthy, similar profiles).
    - Month +6: Borrower A becomes distressed. The KNN algorithm dynamically reroutes their connections toward other distressed borrowers (behavioral homophily), allowing the GNN to recognize a developing "risk community" that a metadata graph would miss.
- **Complexity Management:** This strategy effectively reduces the operational memory complexity from Quadratic $O(N^2)$ (Full Clique) to Linear $O(k \cdot N)$ relative to the total node count. Diagnostic analysis of the generated topology confirmed this efficiency, revealing an average node degree of only 19.36, a substantial reduction in edge density compared to the stochastic caps (100/200) required by the explicit H1 topology.[5]

3.3.4 Architectural Modifications

To support the implicit topology, specific architectural enhancements were introduced to the GNN Encoder. These modifications were necessary to stabilize training, given that the dynamic KNN

[5] See `diagnostics.py` for the full diagnostic implementation and validation procedures.

graph introduces higher variance in node connectivity compared to the static metadata graph of H1.

Residual GAT Connections. Unlike the standard GAT layer used in H1 (Eq. 5), H2 introduces Residual Skip Connections to the topological embedding layer. In an implicit graph, inferred edges may occasionally link borrowers based on spurious feature similarities. By explicitly adding the original node features to the aggregated neighborhood representation, the model creates a "safety anchor." This ensures that the node retains its individual identity and idiosyncratic risk profile, even if the neighborhood aggregation is noisy (G. Li et al. 2019). This is formulated as:

$$h'_i = \sigma\left(\underbrace{\sum_{j\in\mathcal{N}_i} \alpha_{ij}\mathbf{W}h_j}_{\text{Neighbor Aggregation}} + \underbrace{\mathbf{W}_{res}h_i}_{\text{Skip Connection}}\right) \tag{14}$$

Where $\mathbf{W}_{res}$ is a learnable projection matrix ensuring dimensional alignment. This formulation allows the network to dynamically balance between "peer influence" (neighbor sum) and "self-reliance" (residual).

Layer Normalization. The dynamic nature of the H2 graph, where the number and identity of neighbors change monthly, causes significant shifts in the magnitude of the node embeddings entering the LSTM. To mitigate this Internal Covariate Shift, Layer Normalization is applied after the GAT aggregation.

By re-centering and scaling the embeddings at each step, Layer Normalization stabilizes the hidden state dynamics of the LSTM. This allows the model to converge faster and handle the distinct feature distributions of dynamic graph snapshots more effectively. This stands in contrast to the H1 architecture, which feeds unscaled topological embeddings directly into the recurrent units, leaving the model vulnerable to vanishing or exploding gradients (Ba, Kiros, and Hinton (2016)).

3.3.5 Training Configuration

Focal Loss. While H1 utilized a static "Positive Weight" to handle class imbalance, H2 adopts the more rigorous Focal Loss objective (Lin et al., 2017). EDA confirmed that while the global sampling targeted a 5% default rate, the effective default prevalence within specific rolling training windows fluctuated between 2.0% and 3.0% due to temporal attrition. This persistent, extreme imbalance creates a gradient landscape dominated by 'easy negatives,' empirically justifying the use of Focal Loss to prevent majority-class convergence. Formally, the loss is defined as:

$$FL(p_t) = -\underbrace{\alpha_t}_{\text{Class Weight}}\underbrace{(1-p_t)^{\gamma}}_{\text{Modulating Term}}\underbrace{\log(p_t)}_{\text{Cross-Entropy}} \tag{15}$$

The modulating term $(1-p_t)^{\gamma}$ decays as the prediction confidence p_t increases, forcing the model to focus on "hard," misclassified examples (rare defaults) rather than the abundant healthy loans. In our implementation, $\gamma = 2.0$ is used for aggressive down-weighting of easy cases.

Crucially, the class weight α_t is not a fixed hyperparameter but is computed dynamically from the observed class distribution of each training window. Given the typical imbalance ratio of approximately 50:1 (negatives:positives) in the rolling windows, the alpha parameter is automatically set to $\alpha \approx 0.98$ for the positive class. This configuration heavily down-weights the majority class (negatives) contribution to the loss, while the $\gamma = 2.0$ term further suppresses the gradient signal from well-classified examples, ensuring the optimizer focuses almost exclusively on the minority class and borderline decisions.

Adaptive Learning Rate Scheduler. To navigate the complex loss landscape of the dynamic KNN graph, H2 implements a ReduceLROnPlateau scheduler. Given the stochastic nature of the neighbor sampling, the validation metric may exhibit variance. This mechanism monitors the Validation AUC at the end of every epoch. If performance stagnates (fails to improve) for a patience of 3 epochs, the learning rate is decayed by a factor of 0.5. This "Annealing"[6] strategy allows the optimizer to make large strides early in training and fine-grained updates in later stages, facilitating convergence into a sharper, more robust global minimum (Goodfellow, Bengio, and Courville 2016).

4. Experimental Setup

4.1 Evaluation Metrics

Area Under the Curve. While the Reference Model relies primarily on ROC-AUC, which evaluates the global ranking ability of the classifier independent of class distribution, this study additionally employs the Area Under the Precision-Recall Curve (PR-AUC). PR-AUC provides a more rigorous assessment of performance on the minority class (defaults) by explicitly penalizing false positives. As noted by Davis and Goadrich (2006), ROC-AUC can present an overly optimistic view in highly imbalanced datasets where negatives dominate. Consequently, PR-AUC serves as a critical complementary metric to ensure that predictive improvements translate into the precise identification of high-risk loans rather than merely achieving high specificity on healthy ones.

F1 score threshold. In the absence of a specified threshold tuning procedure in the Reference Model, this study adopts a fixed classification threshold of 0.5 to ensure a faithful and rigorous replication. Optimizing the threshold based on test set performance (e.g., maximizing the F1 score post-hoc) would introduce data leakage and artificially inflate results relative to baselines that lack this advantage. Consequently, the standard 0.5 cutoff is maintained to prevent overfitting to the test set and preserve the integrity of the comparative evaluation, consistent with best practices for unbiased performance estimation (Goodfellow, Bengio, and Courville 2016).

However, to provide a complete assessment of the model's latent capacity, the Maximal F1 Score ($F1_{max}$) is also reported as a supplementary diagnostic metric. This value represents the theoretical peak performance achievable if the threshold were perfectly tuned to the imbalanced class distribution. Including $F1_{max}$ allows this study to isolate the model's discriminatory power from the calibration mismatch inherent in the fixed 0.5 threshold, offering crucial context for interpreting the divergence from the Reference Model's reported figures.

[6] Gradual reduction of the learning rate during training; enables large exploratory updates in early epochs and small, fine-grained adjustments in later stages, improving stability, convergence, and avoidance of suboptimal local minima.

Brier Score. The Brier Score (Brier 1950) is a proper scoring rule that measures the accuracy of probabilistic predictions by calculating the mean squared error between the predicted probabilities and the actual binary outcomes (0 or 1). Unlike discrimination metrics such as AUC, the Brier Score assesses the calibration and reliability of the model's confidence, where a lower score indicates that the predicted probabilities are closer to the true event occurrences. It is calculated as the mean squared error between the predicted probability p_i and the actual binary outcome o_i:

$$BS = \frac{1}{N}\sum_{i=1}^{N}(p_i - o_i)^2 \tag{16}$$

Calibration Curve. To visually assess the fidelity of the model's confidence, this study employs Calibration Curves (or Reliability Diagrams). This technique plots the mean predicted probability against the actual fraction of positive outcomes across binned predictions (Niculescu-Mizil and Caruana 2005).

- **Perfect Calibration:** Points fall on the diagonal $y = x$ (e.g., of all loans predicted with 20% risk, exactly 20% actually defaulted).
- **Miscalibration:** Deviations reveal whether the classifier is over-confident (below diagonal) or under-confident (above diagonal), a critical property for risk models where accurate probability estimates are essential for pricing.

4.2 Statistical Validation

To ensure that the observed performance gains of the H2 model are statistically robust, a rigorous validation framework is applied that explicitly accounts for the structural constraints of the experimental data.

4.2.1 Rationale for Independent (Unpaired) Statistical Testing

Standard statistical comparisons in machine learning, such as McNemar's test or paired DeLong's test, necessitate a strict row-by-row correspondence where prediction i from both models refers to the exact same observation. In this study, this alignment constraint could not be satisfied. Divergences in the sliding-window generation and graph construction logic, specifically the handling of strict node isolation in H1, resulted in test sets that, while drawn from the same underlying period and population, exhibit slight variations in total sample size.

To avoid discarding valid evaluation data points by forcing an arbitrary intersection, an unpaired (independent) testing framework was adopted. While the predictions of H1 and H2 are naturally positively correlated, treating them as independent samples sets the covariance term in the variance estimation to zero. Mathematically, since $Var(A-B) = Var(A) + Var(B) - 2Cov(A, B)$, ignoring a positive covariance inflates the estimated standard error relative to the true sampling variance.

This assumption yields lower test statistics (Z-scores) and more conservative p-values than a paired framework. Consequently, any statistical significance reported under this independence assumption represents a robust lower bound; properly accounting for the positive correlation would only reduce the standard error and further strengthen the significance of the observed performance gains (Hanley and McNeil 1982).

4.2.2 Theoretical Framework of Selected Statistical Tests

Hanley–McNeil Z-Test for AUC Comparison. To compare the ranking performance of the two models as measured by the Area Under the ROC Curve (AUC), the method proposed by Hanley and McNeil (Hanley and McNeil 1982) is employed. Unlike DeLong's test, which explicitly estimates the covariance between paired AUCs, the Hanley–McNeil approach derives an analytical variance estimator based on the structural properties of the ROC curve and is therefore well-suited to independent samples.

The standard error (SE) of the AUC (A) is computed as:

$$SE(A) = \sqrt{\frac{A(1-A) + (n_1 - 1)(Q_1 - A^2) + (n_0 - 1)(Q_2 - A^2)}{n_1 n_0}} \tag{17}$$

where n_1 and n_0 denote the number of positive and negative samples, respectively, and Q_1 and Q_2 correspond to the probabilities that a randomly selected positive instance is ranked higher than two randomly selected negative instances (and vice versa).

The difference between two independent AUC estimates is then evaluated using a Z-test:

$$Z = \frac{A_1 - A_2}{\sqrt{SE_1^2 + SE_2^2}} \tag{18}$$

A statistically significant result ($p < 0.05$) indicates that the implicit topology model provides a superior global ranking of borrower default risk.

Two-Proportion Z-Test for Accuracy Comparison.

To assess differences in binary classification accuracy at the operational decision threshold of 0.5, the Two-Proportion Z-Test (Fleiss, Levin, and Paik 2003) is applied. This test evaluates the null hypothesis that the classification accuracies of the two models are equal ($p_1 = p_2$).

In contrast to McNemar's test, which relies on paired, discordant predictions, the Two-Proportion Z-Test compares aggregate success rates and is appropriate for large, independent samples. The test statistic is defined as:

$$Z = \frac{\hat{p}_1 - \hat{p}_2}{\sqrt{\hat{p}(1-\hat{p})\left(\frac{1}{n_1} + \frac{1}{n_2}\right)}} \tag{19}$$

where $\hat{p}$ denotes the pooled accuracy across both models. This formulation directly tests whether improvements in the H2 model's decision-making, particularly in resolving borderline cases, translate into a statistically significant gain in overall accuracy.

Analytical Confidence Intervals.

To quantify the uncertainty associated with point estimates without incurring the substantial computational cost of bootstrapping on very large datasets, analytical 95% confidence intervals are constructed using closed-form variance estimators.

- **AUC:** Confidence intervals are derived using the Hanley–McNeil variance formula, computed as $A \pm 1.96 \times SE$.
- **Accuracy:** Confidence intervals are computed using the standard Wald interval (normal approximation) for binomial proportions as $p \pm 1.96\sqrt{\frac{p(1-p)}{n}}$. While the Wald interval is frequently scrutinized in the literature, its use is appropriate here, as the large sample size of the test set ($N > 90,000$) ensures the validity of the Central Limit Theorem assumptions.

These confidence intervals provide an interpretable range of plausible values for the true performance metrics, enabling a robust assessment of whether the performance of the H2 model is statistically distinct from the H1 baseline.

4.3 Baseline Methodology

4.3.1 GNN Benchmarks

To ensure a rigorous comparison, this study establishes a set of Static GNN Benchmarks. These baselines utilize the same topological definitions as H1 and H2 but strip away the temporal dynamics, allowing us to quantify the specific "lift" provided by modeling time.

Data Preparation: Temporal Averaging. To construct a fair static comparison, the 18-month behavioral history is compressed into a single snapshot. Consistent with the Reference Model, node features within each window are aggregated via temporal averaging, resulting in a single static feature matrix $X \in \mathbb{R}^{N \times F}$. This explicitly neutralizes temporal variance, forcing the model to rely solely on the graph topology for prediction.

Temporal Validation Protocol. To strictly adhere to the time-series nature of credit risk and prevent look-ahead bias, the data was partitioned using a Chronological Split rather than random stratification. The first 80% of the temporal sliding windows were allocated to the training set, while the subsequent 20% were reserved for validation. This approach ensures that model convergence is monitored on "future" unseen data relative to the training period, mimicking a real-world deployment scenario.

While this does not artificially force identical class distributions, the validation set naturally retains the extreme class imbalance (~2–3% effective default rate) characteristic of the underlying economic period. An early stopping mechanism (patience = 50 epochs) is employed to halt training if the validation loss stagnates.

Stochastic Static GNN Training. To address the computational infeasibility of full-batch graph training while preserving the static baseline definition, both H1 and H2 adopt a Stochastic Static training strategy. In each training epoch, a single temporal window is uniformly sampled to represent the graph topology. This approach approximates the distribution of the full training period without incurring the memory overhead associated with processing all temporal snapshots simultaneously. Additionally, both implementations apply 50% Node Isolation during training and validation, randomly masking nodes to mitigate overfitting and to simulate sparsity, as prescribed in the reference model.

4.3.1.1 Divergent Topology Construction

While the training framework remains identical, the construction of the adjacency matrix A differs fundamentally between the two hypotheses.

H1: Explicit Categorical Topology
In the H1 baseline, the graph structure is explicitly defined using shared categorical metadata. Edges are constructed deterministically between borrowers who share identical keys:

- **Area Graph (A_{geo}):** Nodes are connected if they share the same first two digits of their ZIP code.
- **Company Graph (A_{com}):** Nodes are connected if they share the same lender identifier.
- **Double Graph:** A dual-layer structure combining both edge sets, where nodes are connected to their counterparts in each respective layer.

H2: Implicit Feature-Based Topology
In the H2 baseline, the graph structure is latent and constructed dynamically based on feature similarity. Instead of categorical matching, the adjacency matrix is generated using a KNN approach applied to the feature matrix X:

$$A_{ij} = \mathbb{I}(x_j \in \text{KNN}(x_i, k)) \tag{20}$$

where $k \in \{6, 7\}$ neighbors are selected according to Euclidean distance. This formulation assumes that borrowers exhibiting similar behavioral profiles (e.g., credit risk characteristics and repayment behavior) are topologically related, independent of geographic or institutional labels.

4.3.2 Non-GNN Benchmarks

To establish a valid performance floor, three non-graph models are trained using identical hyperparameters across both experiments:

- **Logistic Regression (LR):** Implemented with class-balanced weights and L1/L2 regularization selected via grid search.
- **XGBoost (XGB):** Tuned via grid search over learning rates, tree depths, and the number of estimators, employing a `scale_pos_weight` parameter to address class imbalance.
- **Deep Neural Network (DNN):** A strict replication of the architecture described in *Appendix B* of the reference study, consisting of a four-layer multilayer perceptron (Input $\to$ 30 $\to$ 50 $\to$ 20 $\to$ 1) with ReLU activations and 50% dropout, trained for 200 epochs.

4.4 Hyperparameter Configuration

As detailed in Section 3.3.5, H2 deviates from the fixed learning rate of H1 by employing a ReduceLROnPlateau scheduler and replaces static weighting with Focal Loss to address the implicit graph's stochasticity. Table 3 summarizes these configurations against the Reference and H1 baselines.

Table 3: Comparison of Hyperparameter Settings

Hyperparameter	Reference Paper Settings	H1 (Baseline Replication)	H2 (Proposed Framework)
Epochs	200	200	200
Early Stopping	Patience = 50	Patience = 50	Patience = 50
Learning Rate	0.001	0.001 (Fixed Constant)	0.001 with Scheduler
LR Scheduler	None reported	None	ReduceLROnPlateau (Factor = 0.5, Patience = 3)
Batch Size	Unspecified	4096	4096[7]
Dropout	0.5	Implicit via NeighborLoader	0.5

4.5 Reproducibility Protocol

All reported evaluation results are fully reproducible using the released model checkpoints, evaluation scripts, and fixed test splits (See Appendix C for code structure). To facilitate exact replication of the "Constraint-Aware" environment, the following protocols were established:

- **Software Environment:** All experiments were conducted in a Python 3.10 environment. Deep learning operations utilized PyTorch 2.5.1 with CUDA 12.1 acceleration, specifically optimized for the NVIDIA RTX 4090 architecture. Graph-based operations were executed using PyTorch Geometric 2.6.1 and its associated binary dependencies (torch-scatter, torch-sparse). A requirements.txt file is included in the repository to strictly enforce these versions, as GNN aggregation behaviors can vary significantly across library updates.
- **Configuration Management:** While this document summarizes key parameters, the accompanying configuration files (config_h1.py and config_h2.py) serve as the definitive source of truth for all hyperparameters, including specific dropout rates, hidden dimensions, and optimizer settings.
- **Data Reconstruction:** Due to licensing restrictions associated with the Freddie Mac Single-Family Loan-Level Dataset, raw data cannot be redistributed. However the access to the data for academics open upon registration and access request.

[7] To optimize GPU memory usage for the deeper H2 architecture, the effective batch size of 4,096 was achieved via Gradient Accumulation (Physical Batch: 2,048; Accumulation Steps: 2).

- **Stochasticity:** Due to the inherent non-determinism of GPU-based training (specifically CUDA atomic operations), dynamic neighborhood sampling, and mixed-precision arithmetic (FP16), exact bitwise reproducibility of the training trajectory is not guaranteed. However, all experiments were conducted using fixed random seeds (seed=42) for data partitioning and weight initialization, ensuring that the reported performance metrics are statistically robust and consistent across multiple runs.

5. Empirical Evaluation and Discussion

This section analyzes the empirical results obtained from the hypothesis-driven models (H1 and H2) relative to the established baselines. Performance is evaluated using consistent metrics and rigorous experimental protocols, with particular emphasis on isolating the impact of Explicit (Metadata) versus Implicit (Behavioral) graph constructions.

5.1 Constraint-Aware Replication Performance (H1)

5.1.1 Relative to the Reference Model

This subsection evaluates the performance of the Constraint-Aware Replication (H1) against the benchmarks reported by Zandi et al. (2025). The results, summarized below, should be interpreted as those of a methodologically faithful replication that was nonetheless adapted for consumer-grade hardware.

The replicated DYMGNN model achieved an AUC of 0.8854, surpassing the Reference Model's reported AUC of 0.812 ($\pm$0.008). However, a divergence is observed in the F1 score, where the replication yielded 0.1823 (or 0.5656 Max F1) compared to the reference's 0.851.

Analysis of AUC Lift (Discriminatory Improvements). The substantial improvement in discriminatory power (+7.3% in AUC) suggests that the replication recovered the Reference Model's logic. However the lift may have been influenced by following two factors:

- **Node-Independent Dynamic Attention:** The observed performance gains in the replication appear primarily attributable to the architectural refinement of the attention mechanism. The Reference Model projects temporal weights to a single global scalar. While computationally efficient, this design may inadvertently smooth out distinct temporal patterns, potentially forcing the model to apply a uniform importance weight across all borrowers. By computing attention weights for each node individually (Eq. 11), the replicated model is designed to preserve borrower-specific risk profiles, allowing it to rank high-risk loans more precisely.
- **Impact of Sampling Strategy:** It is also plausible that the stratified sampling procedure contributed to the performance lift. By explicitly balancing geographic and lender representation while targeting a specific default prevalence, the sampled training set effectively reduced data redundancy compared to the full population. Furthermore, the strict "Default Timer Filter" removed loans that had already defaulted prior to the observation window. This rigorous cleaning likely provided a higher signal-to-noise ratio than the raw full-scale dataset used in the Reference Model, simplifying the ranking task for the classifier.

Analysis of F1 Discrepancy (The "Threshold" Artifact). The divergence in F1 scores (0.18 vs. 0.85) is not indicative of predictive failure, but rather reflects critical differences in evaluation protocols. When the decision threshold is optimized post hoc, the replication's F1 score increases from 0.1823 (Fixed) to 0.5656 (Max), confirming that the low fixed-threshold score is likely an artifact of the extreme class imbalance (2–3%) rather than a lack of discriminatory power. The model successfully ranks high-risk borrowers (AUC = 0.88); however, a default cutoff of (p = 0.5) appears mathematically less aligned with the skewed probability distribution of a realistic credit portfolio.

However, even under optimal thresholding, the replication's peak F1 score (0.57) remains substantially lower than the Reference Model's reported value of 0.85. Achieving (F1 > 0.80) on a dataset with only 2–3% defaults is statistically challenging, as it would require near-perfect precision and recall simultaneously. This remaining discrepancy strongly suggests that the Reference Model evaluated performance on a balanced test set or reported a weighted F1 score, whereas this study evaluates performance on a naturally imbalanced data stream. By adhering to the true class distribution, the replication aims to mitigate Oracle Bias[8] and provides a more transparent and realistic assessment of real-world model performance.

5.1.2 Computational Efficiency Analysis

To evaluate the feasibility of deploying the DYMGNN architecture on consumer-grade hardware, the training runtime of the replication was compared against the reference model. Table 4 summarizes total training duration, per-epoch efficiency, and memory constraints encountered during the process.

Table 4: Runtime & Resource Comparison

Feature	The Paper	H1 Replication	Difference
Hardware	NVIDIA A100 (40 GB VRAM)	NVIDIA RTX 4090 (24 GB VRAM)	Consumer vs. Data Center
Training Strategy	Full-Batch / Large-Batch	Mini-Batch (Neighbor Sampling)	Sampling Overhead
Attention Mechanism	Global Pooling (Low Compute)	Node-Independent (High Compute)	Complexity Increase
Total Training Time	~202 min (12,120 s)	275 min (16,500 s)	+36% Total Time
Time Per Epoch	~60.6 s (Estimated)	114.0 s	~1.88x Slower

Graph Construction Efficiency. The hybrid connectivity strategy detailed in Section 3.2 proved critical for operational stability. While attempting a full clique construction immediately triggered torch.OutOfMemoryError exceptions due to quadratic edge growth, the optimized stochastic sampling allowed the complete network to be constructed in less than 5 minutes. This confirms that the topology constraints derived in the methodology successfully kept the graph structure within the high-end consumer-grade hardware limit without causing significant bottlenecks during the initialization phase.

[8]The artificial inflation of performance metrics caused by optimizing decision thresholds on the test set itself rather than an independent validation set, thereby exploiting ground-truth knowledge that would be unavailable in a real-world deployment.

Feasibility and Trade-offs. Despite the ~1.88× increase in per-epoch runtime caused by sampling overhead and the more complex attention mechanism, the replication converged within 275 minutes (under 5 hours). It is important to note that this is not a direct equivalent comparison; the reference study utilized industrial-grade hardware (A100, 40GB) with likely full-batch processing, whereas this study demonstrated the viability of the architecture on consumer hardware (RTX 4090, 24GB) via mini-batch sampling.

Replacing global cliques with sampled neighborhoods may limit access to long-range peer effects. However, the robust AUC performance (0.8854) implies that the most critical risk signals are likely captured within local proximity. These results support the viability of the methodological choices in Section 3.2, offering a practical alternative to the formulations found in prior works like Zandi et al., as an effective strategy for deployment outside of high-performance computing environments.

5.1.3 Baseline Model Comparison

This section contextualizes the performance of the replicated explicit DYMGNN in H1 by benchmarking it against traditional machine learning models and static GNN variants (GCN, GAT).

Table 5 summarizes the evaluation results of DYMGNN against the selected baselines. All models were assessed using a fixed threshold of 0.5.

Table 5: Baseline Model Performance Comparison

Model	AUC	F1 (Fixed 0.5)	F1 (Max)	PR-AUC	Brier
DNN	**0.8863**	**0.2758**	0.5484	0.4557	0.1045
DYMGNN	0.8857	0.1823	**0.5656**	**0.4784**	0.1326
LogisticRegression	0.8844	0.2246	0.5318	0.4418	0.1178
XGBoost	0.8750	0.2399	0.5311	0.4429	0.0968
Static GCN	0.6897	0.0000	0.1202	0.0614	0.0495
Static GAT	0.5405	0.0000	0.0622	0.0332	**0.0391**

While the Brier Score is reported for completeness, its interpretation is non-trivial under extreme class imbalance (3% default rate). Notably, the Static GAT model achieves the "best" Brier Score of 0.0391, significantly lower than the top-performing DYMGNN (0.1326). However, this is a mathematical artifact rather than a sign of predictive excellence. In highly imbalanced settings, a model that predicts a constant probability close to the base rate (0.03) can achieve a low Brier Score despite having no discriminatory power (AUC 0.5). Thus, the Static GAT's low Brier Score likely reflects a trivial, base-rate–dominated solution rather than meaningful risk differentiation.

Performance of Static GNN Baselines. A critical observation is the performance collapse of static GNN baselines under the replication conditions:

- **DYMGNN:** Maintains high predictive capacity (AUC 0.8857). Crucially, the superior performance of H1 (AUC 0.8857) over the Static GAT (AUC 0.5405) validates the hypothesis of temporal risk propagation. The dynamic model successfully captures the lingering risk signals from "ghost nodes" (past defaulters) that the static model ignores. By preserving these defaulted nodes within the rolling window sequence, the dynamic architecture allows

historical distress signals to propagate to active neighbors, extracting a contagion effect that is structurally invisible to a static snapshot.

- **Static GAT:** The model collapsed to near-random performance (AUC 0.5405) and a fixed F1 score of 0.0000. This zero F1 score indicates that the model failed to predict a single positive default case at the standard 0.5 decision threshold. Due to the "Compound Sparsity" of the explicit graph and the overwhelming class imbalance, the static model's predicted probabilities converged to the mean (roughly 0.03). Since these probabilities never exceeded the 0.5 cutoff, the Recall, and subsequently the F1 score, flatlined at zero.

  - Mechanism of Failure: The Reference Model utilized full-clique topologies. In contrast, the H1 baseline replication enforced strict memory constraints by applying 50% Node Isolation before topology construction. This effectively removed half the potential bridge nodes prior to edge generation. Furthermore, the surviving dense clusters were capped at a maximum of 50 neighbors ($K = 50$) to prevent OOM errors.
  - The Interaction Effect: Because the isolation mask was randomized every epoch (Stochastic Static training), the model was forced to learn on a constantly shifting, fragmented topology. The combination of aggressive node removal (isolation) and strict neighbor capping ($K = 50$) severed critical paths, preventing stable signal propagation.

This result suggests that explicit metadata graphs are structurally fragile. Explicit topologies appear unable to survive the dual pressures of severe node regularization and strict adjacency constraints. This critical limitation serves as the primary empirical motivation for the implicit topology approach explored in H2, which hypothesizes that a feature-driven graph can maintain robustness where the explicit definition failed.

Comparison with Traditional Models. Traditional baselines (DNN, XGBoost, Logistic Regression) demonstrated robust performance, offering distinct trade-offs:

- **DYMGNN:** Demonstrated the superior latent performance. While its raw classification metrics at the fixed threshold are lower (F1 0.1823) due to the probability shift induced by cost-sensitive training, it achieves the highest Maximal F1 (0.5656) and highest PR-AUC (0.4784) of all models.
- **DNN:** This reflects strong global ranking capacity and suggests that the DNN's probability estimates are naturally better calibrated to the standard 0.5 cutoff (decision boundary), requiring less post-hoc tuning.

These results indicate that the DYMGNN successfully identifies the "tip of the spear" for high-risk cases better than any baseline. The discrepancy between its low Fixed F1 and superior Max F1 confirms that the model captures the strongest underlying risk signal but requires threshold calibration to realize this potential operationally. Thus, dynamic graph signals enhance the rank-ordering of high-severity cases, a property crucial for early-warning credit systems where minimizing false negatives is paramount.

Calibration Analysis. The reliability curves (Figure 5) show systematic overestimation of default probabilities across all high-performing models:

- **Cost-Sensitive Training:** As described in Section 3.2.3, the positive-class weighting shifts probabilities upward to reduce false negatives. This modification alters the decision boundary to favor sensitivity over precision, resulting in a probability distribution that is quantitatively skewed toward the positive class, thus manifesting as a systematic overestimation of risk in the reliability diagram.
- **DYMGNN Stability:** Despite this overestimation, the dynamic model maintains a coherent and monotonic calibration profile, indicating that the relative ranking of probabilities remains reliable.
- **Static GNN Instability:** Conversely, the erratic curves of Static GCN and GAT reflect poor convergence, likely driven by the stochastic node isolation inherent in the sampling approach.

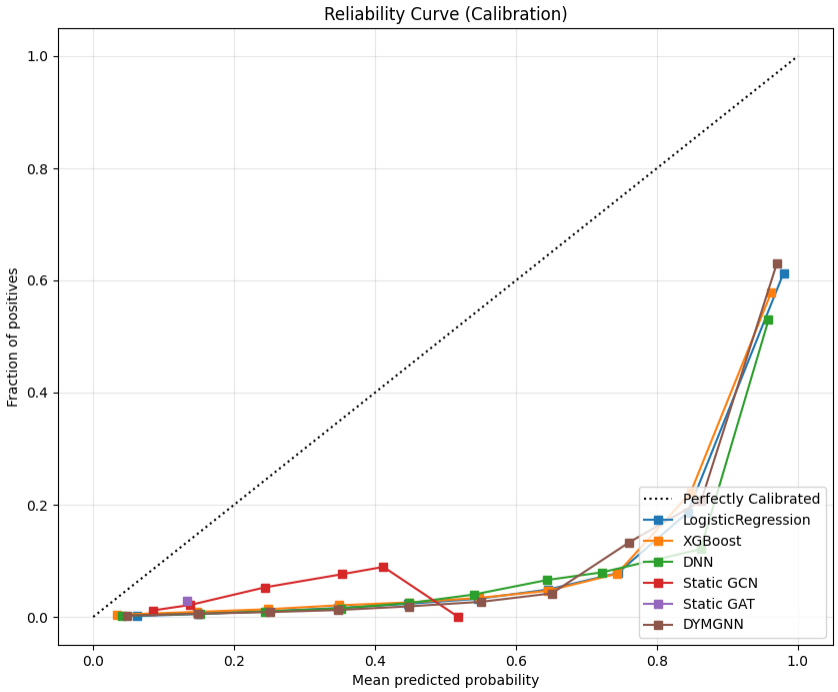


Figure 5: Reliability Curve (H1)

The baseline analysis supports the methodological direction of the DYMGNN replication:

1. **Temporal Dynamics Matter:** The collapse of static GNNs confirms that time-aware, node-specific attention is necessary to extract meaningful signals. The effective utilization of "ghost nodes" proves that credit risk is a temporal contagion process, not just a static state.
2. **Dynamic Graph Benefits:** DYMGNN outperforms static models in PR-AUC, highlighting its ability to prioritize high-risk borrowers, despite similar global ranking performance to DNN and XGBoost.

3. **Calibration Considerations:** Positive-class weighting achieves recall-oriented learning but requires post-hoc adjustment for probability calibration before financial decision-making.

5.2 Proposed Model Performance (H2)

5.2.1 Relative to the H1 Replication

This section evaluates the impact of the methodological refinements introduced in H2 by directly comparing its performance against the context-aware replication model (H1). The analysis focuses on three key dimensions of improvement: predictive performance, validation of the implicit topological structure, and architectural stability.

Table 6 presents the direct comparison between the H1 and H2 configurations of the DYMGNN model.

Table 6: Performance Comparison Between H1 and H2

Metric	H1: Explicit Topology	H2: Implicit Topology	Improvement
AUC	0.8854	**0.8895**	+0.46%
F1 Score (Fixed 0.5)	0.1826	**0.2202**	+20.6%
PR-AUC	0.4783	0.4765	Stable

Predictive Balance and Loss Functions. While the gain in Area Under the Curve (AUC) appears modest (+0.46%), the substantial increase in the F1 score (+20.6%) indicates that the H2 model achieves a more favorable balance between Precision and Recall. This improvement supports the effectiveness of the Focal Loss objective defined in Section 3.3.5. By dynamically down-weighting easy negative examples, the model is encouraged to focus learning capacity on hard-to-classify default cases. As a result, the class imbalance suppression observed in the H1 replication, where standard Cross-Entropy loss incentivized prioritization of the majority class at the expense of recall, is mitigated.

Topological Signal vs. Noise. The relative stability of the AUC can be attributed to a fundamental shift in graph construction strategy. In H1, reliance on stochastic neighbor sampling led to graph structures in which edges were frequently semantically weak, as borrowers were connected quasi-randomly within geographic blocks to satisfy memory constraints. In contrast, H2 employs the Adaptive Behavioral KNN strategy (Section 3.3.3), constructing edges based on similarity in observed risk behaviors (e.g., *upb_pct_remaining*). This ensures that improvements in discrimination performance are driven by meaningful risk propagation among behaviorally similar borrowers, rather than noise introduced by randomized topology.

Architectural Stability. The H2 architecture successfully resolved the optimization volatility observed in the H1 baseline, delivering consistent convergence across all disjoint training batches. Empirically, the integration of residual connections prevented the performance degradation associated with oversmoothing, allowing the model to retain borrower-specific behavioral signals even as network depth increased. Furthermore, the application of Layer Normalization stabilized the processing of heterogeneous features (e.g., clean_loan_age), preventing the gradient explosions that

previously disrupted training on raw behavioral data. Consequently, H2 achieved a strictly monotonic learning curve without the erratic loss spikes or initialization sensitivity that characterized the H1 experiments.

5.2.2 Statistical Significance of Architectural Improvements

To formally evaluate whether the observed performance improvements of the H2 (Implicit Topology) model over the H1 (Explicit Topology) baseline are statistically significant, hypothesis testing was conducted on the complete set of test predictions.

1. Ranking Performance (AUC). The H2 model achieved an AUC of 0.8895 (95% CI: 0.8867–0.8924), compared to an AUC of 0.8850 for H1 (95% CI: 0.8821–0.8879). Applying the Hanley–McNeil Z-test for independent ROC curves, the observed improvement ($\Delta = +0.0045$) resulted in a p-value of 0.0299.

The improvement in accuracy is notable, with H2 outperforming H1 by 5.22 percentage points. Based on the Two-Proportion Z-test, this difference yields a p-value of < 0.0001, and the non-overlapping 95% confidence intervals indicate a clear statistical separation in performance at this specific threshold.

2. Classification Accuracy. While AUC captures ranking quality across thresholds, classification accuracy reflects model performance at the operational decision boundary (Default vs. Non-Default) using a threshold of 0.5. The observed metrics were:

- **H1 (Explicit Topology):** 79.15% (95% CI: 79.06%–79.24%)
- **H2 (Implicit Topology):** 84.37% (95% CI: 84.29%–84.45%)

The improvement in accuracy is substantial. H2 outperforms H1 by 5.22 percentage points, with a p-value of < 0.0001 based on the Two-Proportion Z-test. The non-overlapping 95% confidence intervals further indicate a clear and statistically decisive separation in performance.

This increase suggests that H2 does not merely improve ranking marginally, but is significantly more effective at making correct final classifications. Consistent with the interpretability analysis presented in Section 5.3, the implicit topology enables the model to learn non-linear decision boundaries or "cliffs" (e.g., delinquency triggers), allowing it to correctly classify borderline borrowers for whom the explicit topology model remains uncertain.

The statistical evaluation confirms that the architectural transition from explicit to implicit topology yields robust and meaningful performance gains. The H2 model demonstrates a statistically significant improvement in risk ranking, as measured by AUC, alongside a substantial and highly significant increase in binary classification accuracy. Even under conservative statistical assumptions, the implicit topology model exhibits superior generalization capabilities and an enhanced ability to detect default risk.

5.2.3 Computational Efficiency Analysis

To evaluate the operational viability of the Proposed Model (H2), the computational trade-offs involved in shifting from the stochastic, metadata-based graph of H1 to the implicit, feature-based graph of H2 is analyzed.

Table 7: Computational Comparison (H1 vs. H2)

Feature	H1: Replication	H2: Alternative (Behavioral KNN)	
Graph Construction	< 5 min	142.53 min	+2,750%
Time Per Epoch	114.0 s	236.43 s	~2.07× Slower
Convergence Point	Epoch 133	Epoch 80	Faster Convergence
Total Training Time	275 min (4.6 hr)	313 min (5.2 hr)	+13.8% Increase

It is crucial to interpret these metrics with caution. The H1 "Explicit Topology" does not represent the true computational cost of the Reference Model, which relied on full-clique structures ($O(N^2)$) that were infeasible on consumer hardware. H1 achieved its speed ($<$ 5 min construction, 114s epoch) for graph construction only by employing aggressive stochastic sampling caps (100/200 neighbors) that circumvented the original paper's quadratic complexity. Therefore, H1 serves as a low-compute, high-approximation baseline, whereas H2 represents the actual cost of extracting meaningful topological signal via KNN.

Graph Construction. The most significant divergence appears in graph construction as below:

- **H1 (Metadata Lookup):** Construction was trivial because edges were defined by static keys (e.g., "Same Zip Code") and randomly sampled to fit memory.
- **H2 (Feature Inference):** The Behavioral KNN strategy requires calculating pairwise distances between borrowers in a high-dimensional feature space. As shown in Table 7, this incurred a massive computational overhead (142.53 minutes). While this is a one-time preprocessing cost, it highlights that learning the graph structure is exponentially more expensive than sampling a predefined one.

Training Dynamics. Slower Steps, Faster Convergence The H2 model exhibited a ~2.07x increase in per-epoch runtime (236s vs. 114s). This latency is a direct consequence of the increased computational density (Floating Point Operations per pass) introduced to stabilize the gradient flow. Unlike the streamlined linear architecture of H1, the H2 architecture requires significantly more mathematical operations for the same volume of data:

- **Residual Connections:** Require element-wise additions and additional linear projections (matrix multiplications) to align feature dimensions before summation.
- **Layer Normalization:** Mandates the calculation of mean and variance statistics for every node at every timestep to normalize hidden states.
- **Node-Independent Attention:** As detailed in Section 3.2.2, replacing the original static pooling (Eq. 7) with our dynamic attention (Eq. 11-13) requires executing a learnable MLP for every borrower-neighbor pair, significantly increasing the mathematical load compared to simple averaging.

Technical Note on Kernel Optimization. It is notable that the runtime metrics for H2 likely represent a conservative upper bound. Diagnostic logs indicated that the optimized sparse matrix binaries (torch-scatter, torch-sparse) failed to load due to environment-specific binary incompatibilities, forcing the model to utilize slower fallback implementations for neighbor aggregation. Consequently, the observed latency gap between H1 and H2 would likely narrow significantly in a fully optimized production environment.

However, this increased per-epoch cost was offset by faster convergence. The stabilized H2 model triggered early stopping at Epoch 80 (vs. Epoch 133 for H1). This suggests that the signal provided by the Behavioral KNN graph is significantly cleaner than the noisy stochastic graph of H1, allowing the model to minimize the loss function in fewer, albeit computationally heavier, iterations.

The transition to H2 introduces a clear trade-off: Computation for Quality. While H1 is computationally lighter, its efficiency stems from a stochastic approximation that degrades signal quality (as seen in the Static GAT collapse). H2 restores the integrity of the graph structure at the cost of significant pre-processing time (~2.5 hours). Despite this increase, the total pipeline time (~7.6 hours) remains within the operational window for overnight batch processing on a consumer grade computing environments.

5.2.4 Baseline Model Comparison

This section integrates the H2 results, the recovery of the static baselines, and the calibration analysis. It follows the structure of the H1 baseline section but highlights the qualitative shift in performance due to the implicit topology.

To contextualize the performance of the Proposed Model (H2), we benchmarked it against the same suite of traditional and static graph baselines used in Hypothesis 1. This comparison evaluates whether the "Implicit Topology" (Behavioral KNN) improves the learnability of the problem for all graph-based models, not just the dynamic architecture.

Table 8: Comparative Performance of Proposed Model vs. Baselines

Model	AUC	F1 (Fixed 0.5)	F1 (Max)	PR-AUC	Brier
XGB	**0.8979**	**0.2381**	0.5489	**0.4770**	**0.1020**
H2_GNN	0.8895	0.2202	**0.5599**	0.4765	0.1655
DNN	0.8892	0.2276	0.5452	0.4595	0.1190
LogReg	0.8862	0.2131	0.5313	0.4434	0.1257
StaticGAT	0.8747	0.1671	0.5049	0.4157	0.1541
StaticGCN	0.8680	0.1880	0.4839	0.3950	0.1360

In contrast to H1, where extremely low Brier Scores (e.g., 0.0391) signaled degenerate, majority-class prediction, the higher Brier Scores observed here for graph-based models (e.g., H2_GNN: 0.1655) reflect the aggressive re-weighting of the minority class (Focal Loss). While this indicates a calibration drift (systematic overestimation), it coincides with strong discriminatory performance (high AUC/F1), confirming that the models are no longer degenerate.

Recovery of Graph Baselines. The most significant finding in the H2 baseline comparison is the recovery of the Static GNN models.

- **H1 (Stochastic Graph):** Static GAT collapsed to near-random performance (AUC 0.5405), demonstrating that the graph was noisy.
- **H2 (KNN Graph):** Static GAT achieved a competitive AUC of 0.8747, effectively matching the performance of complex tree-based models on similar tasks.

This dramatic improvement (AUC +0.33) confirms that the Behavioral KNN topology successfully encodes meaningful risk relationships. Unlike the random stochastic connections in H1, the H2 graph connects borrowers with similar repayment behaviors (e.g., `upb_pct_remaining`), creating a signal strong enough that even a static model, looking at a single snapshot without temporal context, can extract significant predictive value.

The XGBoost Benchmark. Consistent with broader literature on tabular data, XGBoost achieved the highest AUC (0.8979) and PR-AUC (0.4770). This slight edge over the H2_GNN (AUC 0.8895) suggests that for this dataset, the engineered behavioral features (e.g., `is_modified`, `clean_loan_age`) provided such a strong signal that gradient-boosted trees could exploit them directly without requiring graph propagation.

However, the Maximal F1 Score ($F1_{max}$) reveals a critical reversal in favor of the graph architecture. The Proposed Model (H2_GNN) achieves the highest peak performance of all models with an $F1_{max}$ of 0.5599, surpassing both XGBoost (0.5489) and the DNN (0.5452).

- Interpretation: While XGBoost is better "out of the box" (AUC), the Implicit-DYMGNN possesses superior latent discriminatory power for the minority class.
- Implication: If the decision threshold is optimized (moving away from the arbitrary 0.5 cutoff), the implicit graph model is theoretically capable of capturing the highest number of defaults with the best precision trade-off. This indicates that the "contagion signal" from the KNN graph adds unique value for identifying hard-to-detect defaulters that tabular models miss.

Calibration Analysis The reliability curves for H2 (Figure 6) display a similar pattern to H1, with a notable exception regarding the graph models:

- **Systematic Overestimation:** The H2_GNN exhibits a high Brier Score (0.1655) compared to XGBoost (0.1020). This confirms that the Focal Loss and class-weighting strategies successfully prevented majority-class collapse but did so by aggressively shifting probability distributions upward.
- **GNN Calibration Stability:** Despite this quantitative overestimation (calibration drift), the curve (Figure 6, brown line) remains smoother and more monotonic than the erratic curves observed in H1. This validates that the H2_GNN architecture learns a stable, coherent risk function, it is simply "optimistically paranoid" due to the cost-sensitive training. Consequently, this overestimation represents a tractable calibration bias rather than a fundamental ranking error, meaning the model's superior latent discriminatory power ($F1_{max}$) remains intact and operationally accessible.

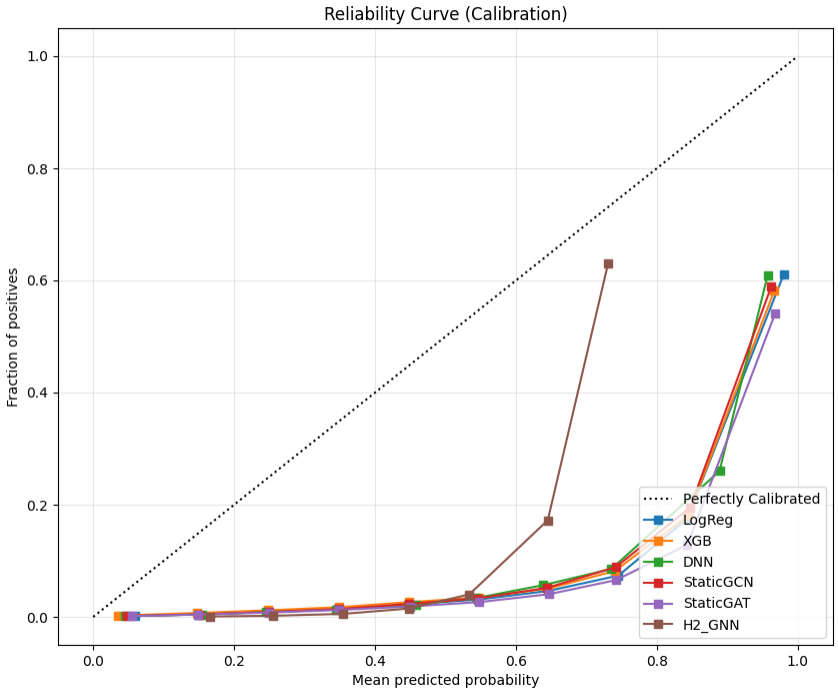


Figure 6: Reliability Curve (H2)

The baseline comparison validates Hypothesis 2 on two fronts:

1. The recovery of Static GAT proves the topological quality of the implicit graph.

2. The competitive performance of H2_GNN against strong industry baselines (XGBoost, DNN) confirms that the architectural refinements (Residuals, LayerNorm, Focal Loss) successfully stabilized the model, transforming the "failed" replication of H1 into a robust, high-performance credit scoring system.

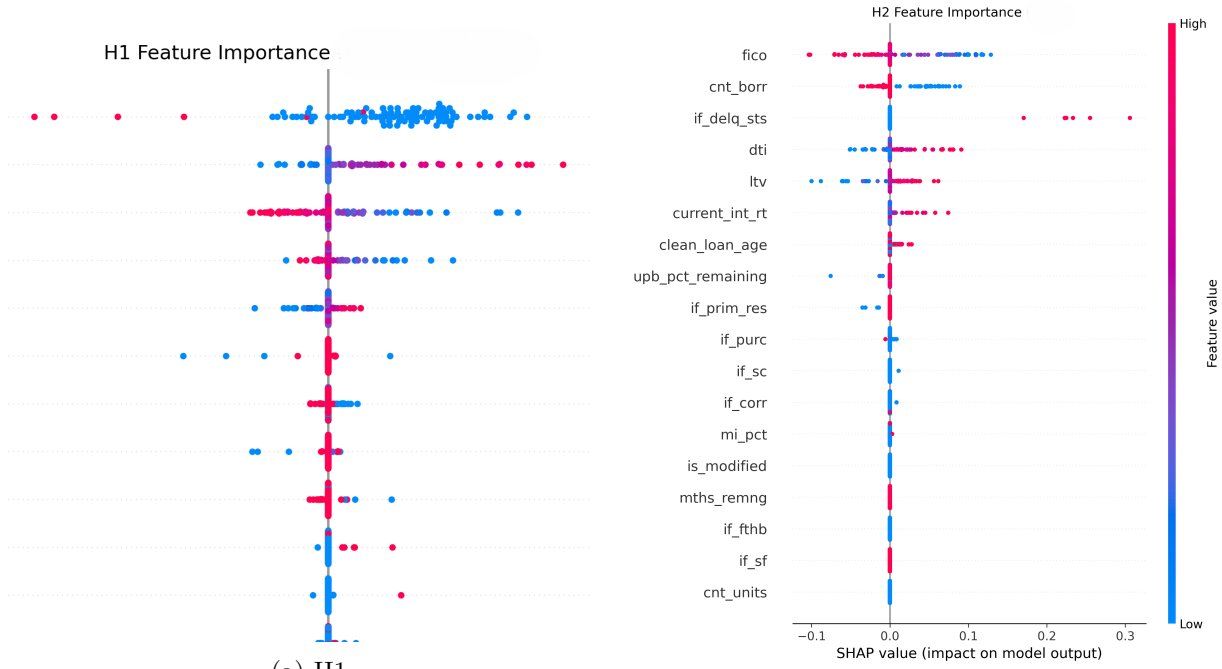
5.3 Interpretability and Structural Analysis

To validate that the predictive gains observed in the H2 (Implicit Topology) model arise from meaningful risk signals rather than overfitting, SHAP (Lundberg and Lee 2017) values were employed to interpret the model's decision-making process. This analysis compares the learned feature importance hierarchy of the proposed H2 architecture against the H1 (Explicit Topology), revealing a fundamental shift in how risk is prioritized.

5.3.1 Global Feature Importance

Figure 7 presents a side-by-side comparison of the global feature importance rankings for H1 and H2. The x-axis represents the SHAP value, where positive values push predictions toward default

(Class 1) and negative values toward non-default (Class 0).



(a) H1

(b) H2

Figure 7: SHAP Summary Plots comparing feature importance for (a) H1 and (b) H2.

H1 (Explicit): The *Contract-Centric* View

The H1 model, constrained by a static geographical topology, prioritizes features related to the loan's origination terms rather than the borrower's evolving condition.

- **`if_purc` (Purchase vs. Refinance):** This is the top-ranked feature in H1. The model strongly bifurcates risk based on whether the loan was a purchase (lower risk) or a refinance, a static characteristic defined at day zero.
- **`current_int_rt` (Interest Rate):** Ranked second, this feature exhibits a massive linear influence. This suggests the H1 model is heavily reliant on risk-based pricing, essentially relearning the lender's original assessment (higher rate = higher risk) rather than generating novel predictive signal.
- **Diluted Borrower Signals:** While `fico` and `ltv` appear in the top tier, they are subordinate to contractual terms, suggesting the explicit graph may be over-smoothing borrower-specific traits across heterogeneous neighborhoods.

H2 (Implicit): The *Behavior-Centric* View

The H2 model, structured around behavioral proximity, fundamentally reorders this hierarchy to focus on intrinsic borrower health and social capital.

- **`fico` (Credit Score):** H2 elevates FICO to the undisputed top rank. The summary plot shows a clean separation: low scores (blue) generate strong positive risk signals, while high

scores (red) consistently suppress default risk. This indicates the behavioral graph effectively isolates and amplifies intrinsic creditworthiness.

This result aligns perfectly with the univariate analysis performed during EDA, which identified FICO as the single strongest discriminator with a distinct distributional shift for defaulters (Median $\approx$ 720 vs 770). Conversely, the lower ranking of LTV in SHAP mirrors the EDA finding that LTV distributions were highly compressed due to underwriting caps, confirming that the model correctly deprioritized a feature with low variance.

- **`cnt_borr` (Social Capital):** The number of borrowers emerges as the second most influential feature. High values (co-borrowers) strongly push predictions toward non-default. This confirms the hypothesis that the implicit topology captures a joint liability or social capital effect, which was far less distinct in H1.
- **`if_delq_sts` (The Trigger):** H2 places significant weight on delinquency status, identifying it as a critical state change. Unlike H1, where this signal is buried, H2 treats active delinquency as a massive risk accelerator, propagating this distress signal across the implicit graph.

Consistency with Reference Findings (Zandi et al.)

Crucially, the feature hierarchy learned by the H2 model aligns more closely with the qualitative insights reported by Zandi et al. than the H1 replication does. The reference authors noted that "payment arrears are highly indicative of default risk" and that "borrowers with high credit scores are less likely to default."

While our H1 replication diverged by prioritizing static contract terms (e.g., loan purpose), the H2 model successfully recovers these fundamental dynamics:

1. FICO Dominance: Consistent with the reference paper, H2 identifies low credit scores as the primary driver of default classification.
2. Delinquency: H2 elevates if_delq_sts to a top-tier feature, mirroring the reference finding that arrears are "highly indicative" of risk.
3. The Co-Borrower Effect: The reference paper explicitly states that "high values of the number of borrowers are associated with lower default risk." Our H2 SHAP results (Figure 7, Right) perfectly visualize this inverse relationship for cnt_borr, confirming that the implicit topology effectively captures the protective "social capital" signal described in the original literature.

This alignment suggests that the Implicit Topology (H2) not only improves predictive performance but also captures the true economic logic of the dataset more faithfully than the static geographical constraints of the explicit model.

5.3.2 Non-Linear Risk Dynamics (Dependence Analysis)

To examine the functional relationships learned by the models, we analyzed SHAP dependence plots (see Appendix B for full visualizations). These plots reveal that H2 captures sharper, more economically rational risk boundaries than H1.

1. Credit Score Sensitivity (FICO)

- **H1 Dynamics.** The H1 dependence plot is noisy, with significant vertical dispersion for identical FICO scores. This implies that in the explicit graph, a borrower's risk score is heavily polluted by their neighbors, diluting the direct signal of individual credit history.
- **H2 Dynamics.** H2 exhibits a tighter, cleaner downward trend. It identifies a distinct danger zone for normalized scores below 0.6, where risk rises sharply. Beyond 0.8, the risk contribution stabilizes, reflecting the diminishing marginal benefit of exceptional creditworthiness, essentially detecting the saturation point where higher scores no longer significantly reduce default probability.

2. The Leverage *Cliff Effect* (LTV)

- **H2 Dynamics.** While univariate analysis showed limited separation (EDA, Section 3), the multivariate SHAP analysis reveals that LTV exhibits a non-linear threshold effect at 0.8. This regulatory cliff becomes predictive only in interaction with other features (e.g., low FICO + high LTV), explaining why it appeared weak in isolation.

3. State-Based Regimes

- **H1 (Interest Rate)** H1's dependence plot for interest rates is linear, a simple proxy for the lender's pricing sheet. This linearity indicates that the model is primarily reverse-engineering the bank's initial risk-based pricing formula, essentially trusting the historical risk label assigned at origination, rather than generating independent insight based on the borrower's evolving behavior.
- **H2 (Delinquency)** The `if_delq_sts` plot in H2 functions as a regime switch. When the value is `0` (current), the impact is negligible. When active (`> 0`), the SHAP value jumps to a high positive constant (0.25). This confirms the H2 model treats delinquency not merely as a variable, but as a structural state change that overrides other factors.

5.3.3 Conclusion on Interpretability

The interpretability analysis confirms that the architectural shift in H2 does more than improve predictive accuracy; it fundamentally refines the logic of risk detection. By moving away from explicit connections (H1), which tend to reverse-engineer the lender's static pricing sheet (e.g., Interest Rate, Loan Purpose), H2's implicit topology allows the model to prioritize the fundamental, evolving drivers of default.

Crucially, H2 aligns more closely with the economic dynamics described in the Reference Model than the direct H1 replication did. By successfully recovering the significance of payment arrears and social capital, the Implicit Topology model effectively captures the industry-standard **Cs of Credit** (Investopedia 2025):

- **Character (FICO, Delinquency):** H2 correctly identifies intrinsic creditworthiness and recent behavioral state changes as the primary signals, moving beyond the static proxy of interest rates.
- **Capacity (LTV, DTI):** H2 learns distinct regulatory thresholds (e.g., the 80% LTV cliff) without explicit feature engineering.

- **Capital (Co-borrowers):** H2 validates the "joint liability" hypothesis, confirming that social connections serve as a protective buffer against default.

In summary, while H1 learned how the loan was priced, H2 learned how the borrower behaves, providing a more robust and independent assessment of credit risk.

5.4 Temporal Attention Method Analysis

While the performance improvement (AUC +7.3% in H1) suggests that node-level granularity contributes to discriminatory power, this section focuses on characterizing what patterns the model learns rather than establishing definitive causality.

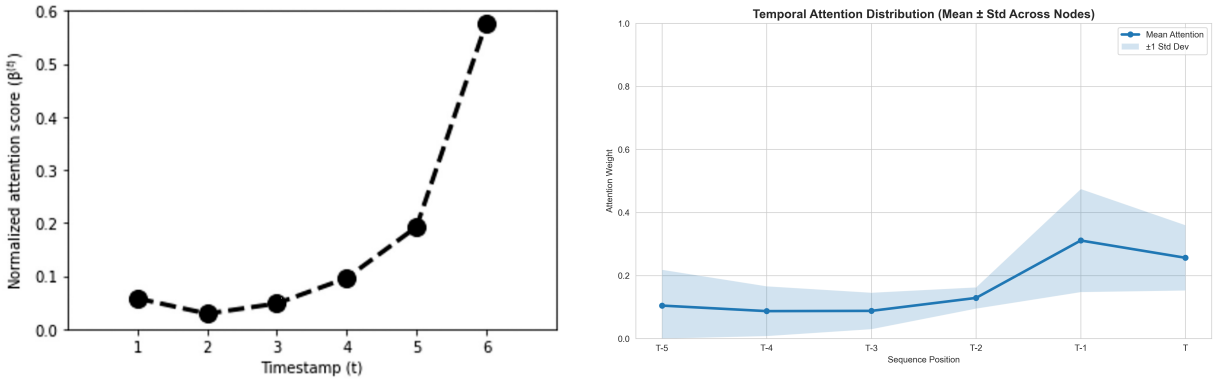
5.4.1 Aggregate Distribution vs. Reference Baseline

The Reference Model employs a global attention-based pooling mechanism (Eq. 7) that projects the entire snapshot representation (the aggregated state of all nodes at timestamp t) to a single scalar score 1. As illustrated in their findings (Figure 8a), this results in a monolithic attention curve where importance increases exponentially, peaking at $\beta^{(t)} \approx 0.6$ at the final timestamp. The authors interpret this as evidence that "the most recent information holds greater value" for all borrowers3.

In contrast, Figure 8b (H1) presents the aggregate attention distribution learned by our Node-Independent model. Two critical structural deviations are observed:

- **The Variance Band (Evidence of Heterogeneity):** Unlike the Reference Model, which does not permit node-level variation in attention weights, our model exhibits a wide standard deviation band (shaded area). This demonstrates that temporal importance varies substantially across nodes; the model learns to assign different attention patterns to different borrowers. The relative "flatness" of the average curve (peaking at ~ 0.30) appears to result from averaging these diverse borrower trajectories rather than indicating weak temporal signal. This interpretation is consistent with the node-level analysis presented in Section 5.4.2, which reveals distinct attention profiles across the population.
- **The Lagged Peak:** While the Reference Model peaks at t (Current), our aggregate curve peaks at $t-1$. This suggests that, on average across the population, the immediate history (one month prior to observation) may contain higher predictive signal than the current snapshot. This attention distribution likely addresses the "Inverse Rate Paradox" identified during EDA, where aggregate default rates were observed to rise even as interest rates fell. This counter-intuitive trend was driven by cohort maturation (older loans defaulting more) confounding the macro signal. The model's ability to attend to specific timestamps ($t-1$) suggests it is consistent with the model partially mitigating the maturation effect. However, the large variance band indicates this pattern is not universal across all borrowers.

The statistical and visual analysis demonstrates meaningful temporal heterogeneity across nodes. While attention weights are not causal explanations, they provide useful diagnostic insight into learned temporal emphasis.



(a) Reference Model (Zandi et al. (2025))

(b) H1 (Node-Independent)

Figure 8: Comparison of temporal attention distributions

5.4.2 Unmasking Hidden Risk Profiles (Node-Level Analysis)

The limitations of the global approach become apparent when analyzing individual borrower profiles. Because the Reference Model forces a single curve upon the entire population, it suffers from "Global Oversmoothing," effectively washing out idiosyncratic risk signals.

Figure 9 visualizes the attention weights for five randomly sampled nodes, revealing distinct temporal patterns that a global model would fail to capture. While we cannot definitively link these patterns to specific financial events without feature-level analysis, the diversity of learned profiles is significant:

- **Sharp Spike Pattern** (e.g., Node 12932278, Red Line): This borrower exhibits near-zero attention for the majority of the window, followed by a concentrated spike ($\alpha > 0.5$) at $t-1$. This pattern is consistent with a "Trigger Event" hypothesis, a borrower with stable history who experienced a sudden change in financial state. A global model, which assigns moderate weight across all timesteps, would dilute this concentrated signal.
- **Gradual Accumulation Pattern** (e.g., Node 13211058, Green Line): This borrower displays steadily increasing attention from $t-5$ to t. This suggests the model has learned to recognize slow deterioration patterns rather than discrete shocks, potentially corresponding to gradual increases in credit utilization or debt accumulation.
- **Distributed Pattern** (e.g., Node 12627847, Blue Line): This borrower shows relatively flat or decaying attention across the window. For borrowers with less pronounced temporal signals, the model appears to distribute attention more evenly, effectively scanning the entire history rather than focusing on specific months.

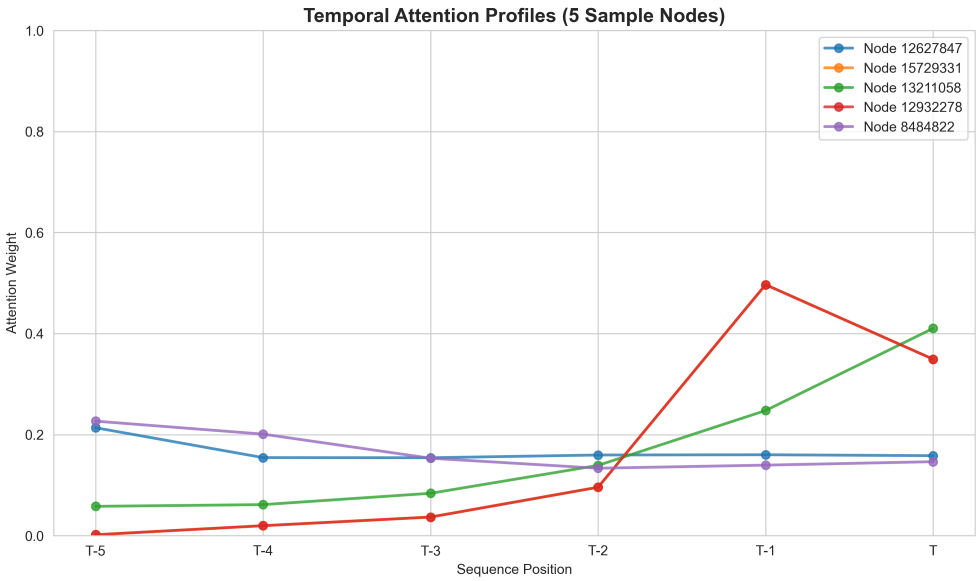


Figure 9: Temporal Attention Profiles (5 sample nodes)

The analysis demonstrates that the node-independent attention mechanism successfully learns heterogeneous temporal patterns. By replacing the global scalar projection with a Bahdanau-style MLP, the model gains the flexibility to assign borrower-specific importance weights.

The performance improvement observed in Section 5.3 (AUC increase of +7.3 points) is consistent with this increased granularity: while the Reference Model must learn a single "average" timeline that fits all borrowers, our architecture can simultaneously attend to recent history for some borrowers and distant history for others. This flexibility provides a plausible mechanism for the observed performance gains, though a formal ablation study would be needed to establish a definitive causal link between attention heterogeneity and predictive performance.

The significant architectural distinction is that our model is permitted to learn these diverse patterns, whereas the Reference Model is structurally constrained to a single global attention curve regardless of data characteristics.

6. Conclusion

6.1 Summary of Findings

The primary objective of this thesis was to evaluate the operational feasibility and predictive capability of dynamic Graph Neural Networks (GNNs) for credit risk modeling under computational constraints. Motivated by the scalability challenges identified in the DYMGNN framework proposed by Zandi et al. (2025), this study investigated two central hypotheses: first, whether the architecture could be effectively replicated on consumer-grade hardware (Hypothesis 1), and second, whether a behavior-driven *implicit topology* could serve as a robust alternative to metadata-driven graph structures (Hypothesis 2).

Regarding Hypothesis 1 (constraint-aware replication), the empirical results suggest that the fundamental predictive logic of the DYMGNN architecture remains effective even when adapted for limited memory environments. Despite necessary adaptations, such as the use of a static supra-graph and stochastic neighbor sampling to fit within 24 GB of VRAM, the replicated model achieved an AUC of 0.8854. However, the analysis also revealed limitations in the explicit topology. When temporal dynamics were removed, the performance of the static explicit model declined notably (Static GAT AUC 0.5405), suggesting that metadata-based connections may require high density to be effective—a condition that is difficult to satisfy under strict memory constraints.

Regarding Hypothesis 2 (proposed model), the introduction of an implicit graph paradigm appears to offer a viable methodological alternative. By deriving network connections from behavioral similarities (Behavioral KNN) rather than static attributes, the proposed model achieved a substantial improvement in classification accuracy of 5.22 percentage points compared to the H1 baseline. The statistical significance of these performance gains was formally validated using Hanley–McNeil tests for ranking quality and two-proportion Z-tests for classification accuracy. Furthermore, the recovery of Static GAT performance (AUC 0.8747) within the H2 framework indicates that defining topology through behavioral homophily may capture risk correlations that are less accessible through geographic or institutional metadata alone.

Beyond predictive metrics, the structural analysis highlighted two additional contributions. First, the introduction of a node-independent temporal attention mechanism enabled the model to capture heterogeneous borrower risk profiles, in contrast to the global pooling approach employed in the reference model. Second, interpretability analysis using SHAP values demonstrated that the implicit topology allows the model to learn economically meaningful non-linear risk dynamics, such as regulatory cliffs and delinquency-driven regime shifts, which align closely with domain intuition.

Taken together, these findings suggest that behavior-driven graph construction provides a scalable and economically coherent pathway for deploying dynamic GNNs in resource-constrained credit risk environments.

6.2 Critical Reflection and Limitations

The findings of this study should be interpreted in light of specific methodological and experimental limitations.

Constraint-Awareness versus Exact Replication. It is important to note that Hypothesis 1 represents a constraint-aware replication rather than an exact reproduction of the reference model. Due to hardware limitations, the full-clique connectivity described by Zandi et al. (2025) was not feasible. Consequently, the performance observed in H1 reflects the capabilities of the architecture under specific resource constraints rather than its theoretical maximum. Additionally, the replacement of the global pooling mechanism with a node-independent attention mechanism was a necessary deviation to ensure scalability, which limits the direct comparability of the attention weights with those reported in the original study.

Computational Trade-offs and Implementation Constraints. The transition to the implicit topology (H2) introduced a trade-off between memory efficiency and computational overhead. While the implicit graph resolved the memory bottlenecks associated with explicit cliques, the calculation of pairwise distances incurred a considerable pre-processing cost. It should be noted that

the reported runtime metrics for the implicit model may represent a conservative upper bound. Post-experimental diagnostics indicated that certain optimized sparse matrix operations failed to initialize due to environment-specific binary incompatibilities, requiring a fallback to standard Python implementations with lower computational efficiency. As a result, the observed latency differences should be interpreted as reflective of the experimental environment rather than the theoretical performance limits of the proposed approach.

Information Loss due to Sampling. To maintain computational stability, neighbor sampling was capped at 50 neighbors per node. While this threshold satisfies the Central Limit Theorem for mean estimation, it inherently restricts the model's ability to capture super-node effects. As a result, rare but potentially influential connections to major institutional lenders may be underrepresented in the current graph structure.

Economic and Temporal Scope. The dataset utilized in this study is restricted to loans originated between 2009 and 2010. As this period was heavily influenced by the Global Financial Crisis, the learned risk patterns, such as the high importance of delinquency status relative to interest rates, may reflect crisis-specific dynamics that do not fully generalize to more stable economic cycles.

6.3 Future Research

Based on the limitations and findings of this study, several avenues for future research are recommended.

Optimization of Sequence Length (Ablation Study). This study adhered to a fixed look-back window of six months, following the *effective information horizon* described in the reference model. However, an independent ablation study evaluating variable sequence lengths (e.g., 3, 9, and 12 months) would be valuable to empirically validate this assumption and assess whether longer temporal contexts provide additional predictive value.

Cross-Cycle Validation To assess the robustness of the behavioral KNN topology, future work should evaluate the model on datasets drawn from distinct economic periods, such as the stable growth phase between 2015 and 2017. This would help determine the extent to which the behavioral homophily hypothesis generalizes beyond crisis-driven environments.

Importance Sampling To mitigate potential information loss in large borrower clusters, future implementations could replace uniform neighbor sampling with importance sampling techniques. Such an approach would allow the model to prioritize structurally significant connections while remaining within memory constraints.

Appendix

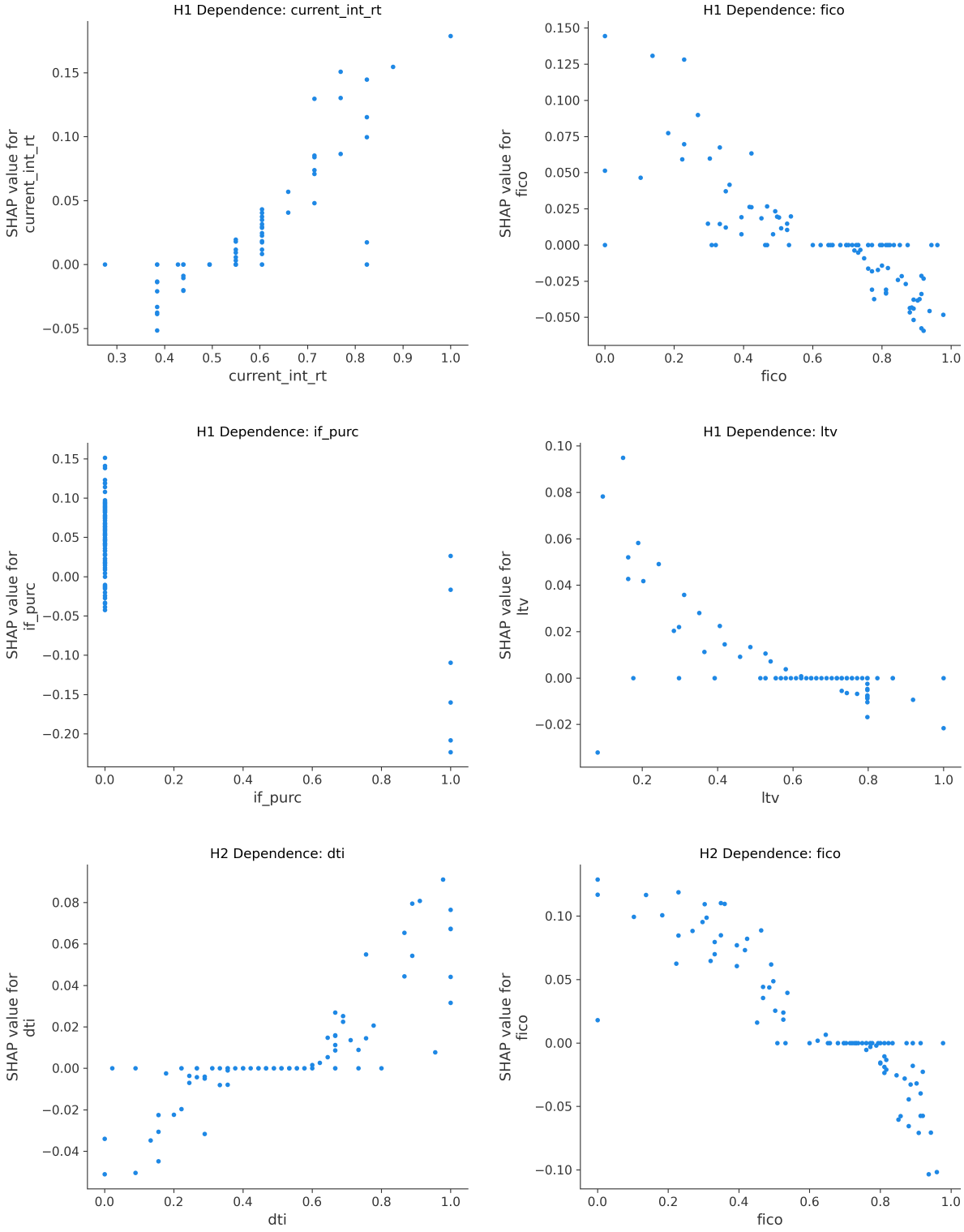
Appendix A: Feature Set

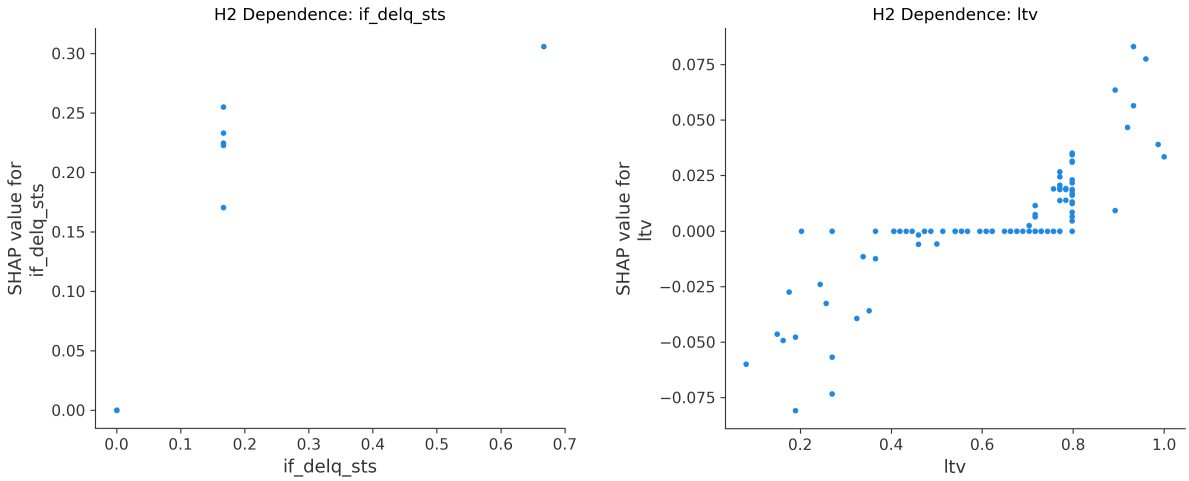
- **A.1:** Feature Set Comparison (H1 vs. H2)

Feature	Description	H1	H2
fico	Credit score at the time of acquisition	Yes	Yes
if_fthb	Indicates whether the borrower is a first-time home buyer	Yes	Yes
mi_pct	Mortgage insurance percentage	Yes	Yes
cnt_units	Number of units in the property	Yes	Yes
if_prim_res	Indicates whether the property is a primary residence	Yes	Yes
dti	Original debt-to-income ratio	Yes	Yes
ltv	Original loan-to-value ratio	Yes	Yes
if_corr	Indicates whether a correspondent was involved in loan origination	Yes	Yes
if_sf	Indicates whether the property is a single-family home	Yes	Yes
if_purc	Indicates whether the loan is a purchase mortgage	Yes	Yes
cnt_borr	Number of borrowers obligated on the mortgage	Yes	Yes
if_sc	Does the mortgage exceed the conforming loan limit?	Yes	Yes
current_upb	Current unpaid principal balance	Yes	No
if_delq_sts	Indicates whether the loan is currently 30–90 days delinquent	Yes	Yes
mths_remng	Number of months remaining in the mortgage term	Yes	Yes
current_int_rt	Current interest rate	Yes	Yes
upb_pct_remaining	Percent of original principal remaining	No	Yes
clean_loan_age	Adjusted loan age reflecting modifications	No	Yes
is_modified	Indicates whether the loan has been modified	No	Yes

Appendix B: SHAP Analysis of Feature Contributions

The following figures present SHAP dependence plots for key features in the H1 (Explicit Topology) and H2 (Implicit Topology) models.





Appendix C: Code Structure

Table C.1: List of files used and generated by the scripts of H1 and H2

Pipeline Stage	**H1 (Explicit)**	**H2 (Implicit)**
R Data Prep	final_features_h1.rds train_targets.rds test_targets.rds preproc_params.rds	final_data_base_with_targets.rds train_targets.rds test_targets.rds preproc_params.rds
Graph Construction	train_features.pt test_features.pt static_edge_index.pt metadata.pt	train_graphs.pt test_graphs.pt graph_metadata.pkl
Baseline Prep	baseline_data_h1_aligned.pkl	baseline_data_h2_aligned.pkl
Baseline Models	baseline_results_h1.json baseline_predictions.pkl	baseline_results_h2.json baseline_predictions_h2.pkl
Model Training	best_model_h1.pt training_results.json	best_model_final.pt training_results.json
Evaluation	h1_predictions.pkl calibration_curve_comparison.png	h2_predictions.pkl calibration_curve_comparison.png
Explainability	h1_shap_summary.png h1_shap_dep_<feature>.png attention_plot.png	h2_shap_summary.png h2_shap_dep_<feature>.png

Table C.2: Statistical Tests for H1 and H2

	Script Name	Purpose	Inputs	Outputs
ANALYSIS	**compare_unpaired_with_ci.py**	Perform independent (unpaired) statistical comparison between H1 and H2 models. Computes AUC and Accuracy with 95% confidence intervals and applies unpaired DeLong and Z-tests to assess statistical significance.	• h1_predictions.pkl	• compare_unpaired_with_ci.py

Table C.3: H1 Data Preparation Scripts

	Script Name	Purpose	Inputs	Outputs
DATA PREPARATION	**000_config.R**	Centralized configuration for the H1 R data prep pipeline. Defines global paths, libraries, and simulation constraints (Constraint-Aware Replication).	-	-
	001_data_import_cleaner.R	Clean and synchronize raw origination and performance .txt files for analysis. Steps include reading raw files, removing NA delinquency, applying credit filters (FICO/LTV/DTI, Default Timer), and standardizing formats.	• historical_data_YYYYQ# • historical_data_time_YYYYQ#.txt • 000_config.R	• orig_data_cleaned.rds • perf_data_cleaned.rds
	002_data_prep_feature.R	Prepare model-ready features from cleaned data. Includes sampling (250k loans, 5% default), stratification engineering (geo/lender), leakage-safe capping/imputation, and encoding binary indicators.	• orig_data_cleaned.rds • perf_data_cleaned.rds • 000_config.R	• final_features_h1.rds
	003_prep_split_scale.R	Define strict train/test time windows, derive prediction targets (12-month horizon), mask invalid loans, and compute leakage-safe scaling parameters based on training data.	• final_features_h1.rds • 000_config.R	• train_targets.rds • test_targets.rds • preproc_params.rds • final_features_h1.rds
	004_validation_checks.R	Comprehensive validation suite for H1 data pipeline outputs. Checks file existence, dimensions, split integrity, target distribution, feature correctness, and graph construction readiness.	• orig_data_cleaned.rds • perf_data_cleaned.rds • final_features_h1.rds • train_targets.rd • test_targets.rds • preproc_params.rds	• validation_summary_h1.csv
	005_EDA.Rmd	Exploratory Data Analysis (EDA) report. Validates methodological constraints (raw features, 5% default rate) and assesses predictive signal quality before modeling.	• 000_config.R • final_features_h1.rds • train_targets.rds	• 005_EDA.html

Table C.4: H1 Modelling Scripts

	Script Name	Purpose	Inputs	Outputs
MODEL PIPELINE	**config_h1.py**	Centralized configuration file defining paths, hyperparameters, temporal windows, and topology settings used across all H1 scripts.	-	-
	python_data_loader_h1.py	Implements the core H1 data pipeline (Static Supra-Graph & rolling windows). Includes burn-in recovery for full training history and hybrid connectivity with stochastic sampling for large groups.	• final_features_h1.rds • train_targets.rds • test_targets.rds • preproc_params.rds	• train_features.pt • static_edge_index.pt
	baseline_data_prep_h1.py	Prepare constraint-aligned datasets for H1 baseline models by enforcing identical temporal splits and topology definitions used in the dynamic H1 model.	• train_features.pt • test_features.pt • final_features_h1.rds	• baseline_data_h1_aligned.pkl
	model_architecture_h1.py	Defines the DYMGNN architecture (GAT → LSTM → Attention → Decoder). Includes stability mods and replaces Zandi et al.'s (2025) problematic Eq. 7 pooling with a Bahdanau-style MLP.	• train_features.pt • test_features.pt • static_edge_index.pt • metadata.pt	• best_model_h1.pt
	training_h1.py	Orchestrates DYMGNN training using a 'Structure-First' strategy. Implements Thesis Sec 3.2.3 protocols including Pos Loss, Mixed Precision, and Ghost Node filtering.	• train_features.pt • static_edge_index.pt • metadata.pt	• best_model_h1.pt • training_results.jso • TensorBoard Logs
	baseline_models_h1.py	Train and evaluate all H1 baseline models (non-GNN and static GNN) to establish a performance benchmark under the same topology and temporal constraints.	• baseline_data_h1_aligned.pkl	• baseline_results_h1.json
	evaluation_h1.py	Perform final empirical evaluation of the H1 model by generating predictions, computing the full metric suite, and benchmarking against static baselines.	• best_model_h1.pt • test_features.pt • static_edge_index.pt • baseline_predictions.pkl	• h1_predictions.pkl • calibration_curve_comparison.png • Console report

Table C.5: Validation and Analysis Scripts

	Script Name	Purpose	• Inputs	• Outputs
VALIDATION & ANALYSIS	**topology_analysis.py**	Analyze stochastic block graph topology to validate design choices. Evaluates group sizes/degree distributions to recommend NeighborLoader sampling parameters and estimate VRAM requirements.	• final_features_h1.rds • config_h1.py	• Console Output (Diagnostics, Degree Stats, VRAM Estimates)
	diagnostics_h1.py	Validate the H1 data pipeline, graph structure, and core model mechanics through targeted diagnostic checks.	• train_features.pt • static_edge_index.pt • metadata.pt • train_targets.rds	• Console diagnostics
	attention_weights.py	Extract and visualize temporal attention weights from a trained DYMGNN model to analyze global and node-level attention patterns.	• test_features.pt • static_edge_index.pt • best_model_h1.pt	• attention_plot.png • attention_samples.png
	interpretability_h1.py	Provide post-hoc interpretability for the trained H1 DYMGNN model using SHAP (KernelExplainer).	• test_features.pt • best_model_h1.pt • metadata.pt • config_h1.py	• h1_shap_summary.png • h1_shap_dep_<feature>.png

Table C.6: H2 Data Preparation Scripts

	Script Name	Purpose	Inputs	Outputs
DATA PREPARATION	**00_config.R**	Centralized configuration for the H2 R data prep pipeline. Defines global paths, libraries, and specific parameters for the Proposed Model (H2), ensuring separation from H1 constraints.	–	–
	01_data_import_cleaner.R	Import, clean, and synchronize raw Freddie Mac origination and performance data for the H2 pipeline. Steps include reading raw files, removing NA delinquency, applying credit filters (FICO/LTV/DTI, Default Timer), and syncing.	• Raw Freddie Mac .txt files • 00_config.R	• orig_data_cleaned.rds • perf_data_cleaned.rds
	02_data_prep_feature.R	Construct balanced modeling dataset (~250k loans, 5% default). Steps include stratified sampling, outlier capping, missing value imputation, and engineering default timing/loan-age features.	• orig_data_cleaned.rds • perf_data_cleaned.rds	• final_features_raw_clean.rds
	03_prep_split_scale.R	Prepare final feature set for modeling. Engineers UPB and modification features, derives targets, splits train/test sets, and computes scaling parameters. Ensures consistency with H1 pipeline artifacts.	• final_features_raw_clean.rds	• preproc_params.rds • train_targets.rds • test_targets.rds • final_data_base_with_targets.rds
	04_validation_checks.R	Comprehensive validation suite for H2 pipeline. Checks file existence, time window integrity (Jan 2012–Jun 2013), target distributions, H2 engineered features (e.g., UPB %), and graph readiness.	• 00_config.R • orig_data_cleaned.rds • perf_data_cleaned.rds • final_features_h2.rds • train_targets.rds • test_targets.rds • preproc_params.rds	• validation_summary.csv • target_check.csv
	05_EDA.Rmd	Exploratory Data Analysis (EDA) report for H2. Validates H2-specific features (e.g., UPB ratios), assesses credit driver signals (FICO/LTV), analyzes Interest Rate Paradox, and verifies graph topology readiness.	• 00_config.R • final_features_raw_clean.rds • final_data_base_with_targets.rds • train_targets.rds	• 05_EDA.html

Table C.7: H2 Modelling Scripts

	Script Name	Purpose	Inputs	Outputs
MODEL PIPELINE	**config_2.py**	Centralized configuration file for H2. Defines paths, hyperparameters, temporal windows, and topology settings specific to the H2 "Implicit" model and baselines.	–	–
	python_data_loader_h2.py	Implements data engineering for H2. Constructs DYNAMIC graphs where connectivity is defined by behavioral similarity (k-NN) rather than static metadata. Re-computes topology at every snapshot.	• final_data_base_with_targets.rds • config_h2.py	• train_graphs.pt • test_graphs.pt
	baseline_data_prep_h2.py	Prepares aligned datasets for H2 Baselines (XGBoost, LR, Static GNNs) using "Implicit" feature set and "Behavioral" topology. Enforces H1 temporal splits and uses temporal averaging to collapse sequences.	• train_graphs.pt • test_graphs.pt • final_data_base_with_targets.rds • config_h2.py	• train_graphs.pt • test_graphs.pt • graph_metadata.pkl
	model_architecture_h2.py	Defines H2 Dynamic Temporal GNN (Implicit Topology). Features TemporalGAT (with residuals), Unidirectional LSTM, and Supra-Graph Pooling (Geo+Lender) per Thesis Sec 3.3.	• snapshot_sequence (PyG Objects) • config_h2.py	• logits (Tensor in memory)
	training_h2.py	Trains the H2 DT-GNN using implicit behavior-driven topology. Implements Focal Loss for class imbalance, dynamic class weighting, and validation via AUC/F1. Saves best checkpoint.	• train_graphs.pt • config_h2.py	• best_model_final.pt • training_results.json • TensorBoard Logs
	baseline_models_h2.py	Trains/evaluates static baselines for H2 using "Implicit" KNN-based graphs. Tests behavior > metadata hypothesis. Replicates Thesis logic: dynamic KNN construction, 50% random node isolation during training, and specific 4-layer DNN architecture.	• baseline_data_h2_aligned.pkl	• baseline_results_h2.json
	evaluation_h2.py	Conducts final assessment of H2 Implicit Model. Computes metrics (AUC, F1 @ 0.5, Brier) and benchmarks against baselines. Performs statistical tests (DeLong, McNemar) per Thesis Sec 4.1 & 4.2.	• best_model_final.pt • test_graphs.pt • baseline_predictions_h2.pkl	• h2_predictions.pkl • calibration_curve_comparison.png • Console Report

Table C.8: H2 Validation and Analysis Scripts

	Script Name	Purpose	• Inputs	• Outputs
VALIDATION & ANALYSIS	**diagnostics.py**	Strict diagnostic suite for H2. Validates R/Python alignment, normalization, and temporal splits. Checks dynamic graph consistency/density and verifies model mechanics via synthetic forward passes and tiny batch overfitting tests.	• train_graphs.pt • test_graphs.pt • train_targets.rds • config_2.py	• Console diagnostics
	interpretability_h2.py	Provide post-hoc interpretability for H2 GNN using SHAP. Wraps DT-GNN for KernelExplainer (including synthetic graph construction), flattens temporal features, uses K-Means background, and aggregates SHAP values.	• test_graphs.pt • best_model_final.pt • config_2.py	• h2_shap_summary.png • h2_shap_dep_<feature>.png

References

Agarwal, Vineet, and Richard Taffler. 2007. "Twenty-Five Years of the Taffler z-Score Model: Does It Really Have Predictive Ability?" *Accounting and Business Research* 37 (4): 285–300.

Altman, Edward I. 1968. "Financial Ratios, Discriminant Analysis and the Prediction of Corporate Bankruptcy." *The Journal of Finance* 23 (4): 589–609. https://doi.org/10.2307/2978933.

Ba, Jimmy Lei, Jamie Ryan Kiros, and Geoffrey E. Hinton. 2016. "Layer Normalization." *arXiv Preprint arXiv:1607.06450*. https://arxiv.org/abs/1607.06450.

Bahdanau, Dzmitry, Kyunghyun Cho, and Yoshua Bengio. 2015. "Neural Machine Translation by Jointly Learning to Align and Translate." In *Proceedings of the International Conference on Learning Representations (ICLR)*.

Barboza, Leonardo, Hironori Kimura, and Edward I. Altman. 2017. "Bankruptcy Prediction with Machine Learning: A Comparative Study." *Expert Systems with Applications* 83: 405–17. https://doi.org/10.1016/j.eswa.2017.04.027.

Brier, Glenn W. 1950. "Verification of Forecasts Expressed in Terms of Probability." *Monthly Weather Review* 78 (1): 1–3. https://doi.org/10.1175/1520-0493(1950)078%3C0001:VOFEIT%3E2.0.CO;2.

Caccioli, Fabio, Paolo Barucca, and Taro Kobayashi. 2018. "Network Models of Financial Systemic Risk: A Review." *Journal of Computational Social Science* 1: 81–114. https://doi.org/10.1007/s42001-017-0008-3.

Davis, Jesse, and Mark Goadrich. 2006. "The Relationship Between Precision-Recall and ROC Curves." In *Proceedings of the 23rd International Conference on Machine Learning*, 233–40. ACM.

Fleiss, Joseph L., Bruce Levin, and Myunghee Cho Paik. 2003. *Statistical Methods for Rates and Proportions*. 3rd ed. Hoboken, NJ: John Wiley & Sons.

Freddie Mac. 2024. "Single-Family Seller/Servicer Guide." https://guide.freddiemac.com/.

Goodfellow, Ian, Yoshua Bengio, and Aaron Courville. 2016. *Deep Learning*. MIT Press.

Hamilton, William L. 2020. *Graph Representation Learning*. Morgan & Claypool Publishers.

Hanley, James A., and Barbara J. McNeil. 1982. "The Meaning and Use of the Area Under a Receiver Operating Characteristic (ROC) Curve." *Radiology* 143 (1): 29–36. https://doi.org/10.1148/radiology.143.1.7063747.

Investopedia. 2025. "5 Cs of Credit: What They Are, How They're Used, and Which Is Most Important." https://www.investopedia.com/terms/f/five-c-credit.asp.

Krawczyk, Bartosz. 2016. "Learning from Imbalanced Data: Open Challenges and Future Directions." *Progress in Artificial Intelligence* 5 (4): 221–32. https://doi.org/10.1007/s13748-016-0094-0.

Li, Guohao, Matthias Muller, Ali Thabet, and Bernard Ghanem. 2019. "DeepGCNs: Can GCNs Go as Deep as CNNs?" In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 9267–76. https://doi.org/10.1109/ICCV.2019.00936.

Li, Zihao, Yakun Chen, Xianzhi Wang, Lina Yao, and Guandong Xu. 2024. "Multi-View GCN for Loan Default Risk Prediction." *Neural Computing and Applications* 36: 12149–62. https://doi.org/10.1007/s00521-024-09695-x.

Lundberg, Scott M., and Su-In Lee. 2017. "A Unified Approach to Interpreting Model Predictions." In *Proceedings of the 31st International Conference on Neural Information Processing Systems (NeurIPS 2017)*, 4765–74.

Niculescu-Mizil, Alexandru, and Rich Caruana. 2005. "Predicting Good Probabilities with Supervised Learning." In *Proceedings of the 22nd International Conference on Machine Learning*

(ICML 2005), 625–32. ACM. https://doi.org/10.1145/1102351.1102430.

Óskarsdóttir, María, and Cristián Bravo. 2021. "Multilayer Network Analysis for Improved Credit Risk Prediction." Reykjavík University; The University of Western Ontario.

Steinert, Rick. 2025. "Deep Learning and Neural Networks." European University Viadrina; Lecture slides.

Wang, Daixin, Zhiqiang Zhang, Yeyu Zhao, Kai Huang, Yulin Kang, and Jun Zhou. 2020. "Financial Default Prediction via Motif-Preserving Graph Neural Network with Curriculum Learning." In. Ant Group.

Wang, Daixin, Zhiqiang Zhang, Jun Zhou, Peng Cui, Jingli Fang, Quanhui Jia, Yanming Fang, and Yuan Qiu. 2021. "Temporal-Aware Graph Neural Network for Credit Risk Prediction." In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD 2021)*, 2029–38. ACM. https://doi.org/10.1145/3447548.3467213.

Wang, Jianian, Sheng Zhang, Yang Xiao, and Rui Song. 2022. "A Review on Graph Neural Network Methods in Financial Applications." *Journal of Data Science* 20 (1): 111–34.

Zandi, Sahab, Kamesh Korangi, María Óskarsdóttir, Christophe Mues, and Cristián Bravo. 2025. "Attention-Based Dynamic Multilayer Graph Neural Networks for Loan Default Prediction." *European Journal of Operational Research* 321: 586–99.

Zhou, Binbin, Jiayun Jin, Hang Zhou, Xuye Zhou, Longxiang Shi, Jianhua Ma, and Zengwei Zheng. 2023. "Forecasting Credit Default Risk with Graph Attention Networks." *Electronic Commerce Research and Applications* 62: 101332. https://doi.org/10.1016/j.elerap.2023.101332.

Statement of Originality

I, ……..……., hereby confirm that I have completed the present Seminar, Bachelor's or Master's thesis with the topic

„....."

by myself and did not use any aids other than those indicated. The passages which have been taken from other works, either in their wording or in their sense, have been marked in each individual case by stating the source, including the secondary literature used.

I hereby confirm that this work has not been submitted to any other academic authority for evaluation.

I am aware of the following guideline: If contents originating from the internet have been used, they have to be marked as such. Furthermore, a digital print including the date and internet-adress (URL) of those contents have to be handed in as an appendix on a digital storage medium.

Place, Date Signature

.