Adaptive GPU-Accelerated Graph Learning for Systemic Risk Assessment in Interconnected Financial Markets

Henrietta Emilsson
Department of Computer Science, University of New Hampshire, Durham, NH, USA.
henrietta873@unh.edu

Anita Sellers
Department of Computer Science and Engineering, University at Buffalo, Buffalo, NY, USA.
anitawork@buffalo.edu

Amadeo Nievo
Department of Electrical Engineering and Computer Science, University of Missouri, Columbia, MO, USA.
nievo1983@missouri.edu

Indira Toor
Department of Computer Science, Binghamton University, Binghamton, NY, USA.
indira.toor@binghamton.edu

Abstract

Systemic risk in interconnected financial markets is shaped by time-varying exposures, common holdings, funding dependence, and correlated stress that propagate across institutions and market segments. Conventional connectedness measures and network risk models capture important channels of contagion, but they typically rely on static or low-frequency network representations and struggle to combine heterogeneous relational information with scalable inference. This paper develops a methodological framework for adaptive GPU-accelerated graph learning that treats financial markets as dynamic, typed, and weighted graphs and estimates node-level distress and system-level loss through temporal message passing. The central contribution is an integrated design that couples dynamic graph construction, localized graph convolution with recurrent memory, risk-sensitive learning objectives, covariance-based exposure features, and calibrated predictive uncertainty. The framework specifies how GPU execution can support repeated sparse message passing, adaptive neighborhood sampling, and workload-balanced edge processing while preserving the economic semantics of financial linkages. Analytical assessment focuses on the learning objective, propagation operator, computational complexity, calibration logic, and domain-specific failure modes. The design clarifies when graph learning adds value over econometric and network-based benchmarks, how uncertainty and explainability can be incorporated without sacrificing scalability, and which implementation conditions are necessary for supervisory and risk-management use. The resulting contribution is a rigorous computational and analytical foundation for systemic risk assessment rather than a claim of empirical superiority.

Keywords

systemic risk; graph neural networks; GPU acceleration; financial networks; uncertainty calibration; dynamic graph learning

1. Introduction

Systemic risk describes the possibility that distress in one institution, market, or funding channel becomes amplified through interconnected exposures and produces disproportionate losses across the financial system. The analytical problem is not simply to forecast individual defaults or asset returns. It is to represent the conditional dependence structure among heterogeneous financial entities, estimate how localized shocks propagate through that structure, and distinguish ordinary co-movement from fragile connectivity that can accelerate loss cascades. Empirical finance has produced influential connectedness measures based on return and volatility spillovers [1], variance decompositions [2], and network amplification mechanisms [3]. Network-based risk measures such as DebtRank emphasize that centrality alone is insufficient because distress can propagate through recursive exposure pathways [4]. Broader ecosystem perspectives stress that diversification, complexity, and concentration can produce non-monotonic stability effects [5], while reviews of financial network models show that results depend strongly on the assumed topology, exposure mapping, and shock-propagation rule [6].

These traditions provide the economic foundation for systemic risk assessment, but they leave a computational and representational gap. Financial markets generate relational data at multiple frequencies: balance-sheet exposures, funding relationships, common asset holdings, derivative positions, payment flows, market co-movement, and liquidity linkages. The relevant graph is not a single fixed adjacency matrix. It is dynamic, multilayer, partially observed, and heterogeneous in edge meaning. A static network model may obscure changes in connectivity around stress episodes, while a purely correlation-based model may conflate common shocks with genuine transmission. Graph learning offers a way to learn nonlinear propagation functions directly from node features and relational structure, but a generic graph neural network is not sufficient for systemic risk. The architecture must preserve economically meaningful edge types, incorporate temporal dependence, estimate tail-relevant system loss, express uncertainty, and remain computationally tractable as the number of nodes and temporal graph snapshots grows.

This paper addresses that gap by specifying an adaptive GPU-accelerated graph learning framework for systemic risk assessment. The main contribution is not a single model layer but a coherent methodological design that connects financial graph construction, temporal message passing, risk-sensitive objectives, covariance-based exposure features, calibration, explainability, and hardware-aware execution. The framework treats each market snapshot as a typed graph whose nodes represent financial institutions, assets, or market venues, and whose edges encode exposures, funding dependence, common holdings, or statistical dependence. Edge activity and feature distributions evolve over time, so the learning system adapts neighborhoods and sampling intensity rather than assuming a fixed graph. GPU acceleration is used where the computational burden is concentrated: sparse neighborhood aggregation, repeated temporal propagation, adaptive sampling, and batch construction.

The paper makes four connected arguments. First, systemic risk assessment requires a dynamic and typed graph representation in which propagation mechanisms are separated from statistical co-movement. Second, temporal graph learning can combine local exposure

features with higher-order neighborhood information, but only if the objective explicitly penalizes unstable attribution and rewards calibrated distress probabilities. Third, GPU acceleration is not an implementation afterthought: sparse graph workloads create memory-access, load-imbalance, and kernel-launch costs that shape which adaptive sampling and message-passing designs are feasible. Accelerator-oriented GCN designs show that dataflow and memory hierarchy choices are central to graph convolution performance [12], which implies that a systemic risk model must be co-designed with its execution strategy. Fourth, explainability and uncertainty are functional requirements for risk management, not optional post-processing, because an uncalibrated systemic risk score can mislead supervisory judgment even when it is computationally efficient.

The remainder of the paper is organized as follows. Section 2 formulates the financial-market graph and defines the risk target. Section 3 develops the adaptive GPU-accelerated graph learning architecture. Section 4 specifies risk estimation, uncertainty, and calibration logic. Section 5 presents the methodological framework and analytical assessment, including the learning objective, propagation semantics, computational scaling, evaluation protocol, and failure modes. Section 6 discusses comparative trade-offs and implementation conditions. Section 7 concludes.

2. Problem Formulation

2.1 Financial-Market Graph Representation

The assessment problem begins with a time-indexed graph. Let V denote the set of financial entities under consideration, including banks, insurers, asset managers, central counterparties, and, where relevant, market venues or asset classes. At each time t, edges E_t describe relations among entities. The graph may be directed, because an exposure from i to j need not equal the exposure from j to i, and it may be typed, because funding dependence, common holdings, and derivative exposures carry different propagation semantics. A compact representation is

$$G_t = (V, E_t, A_t, X_t)$$

where G_t is the graph snapshot at time t, V is the node set, E_t is the edge set, A_t is the weighted adjacency structure, and X_t is the node-feature matrix. The adjacency A_t may be stored as a sparse matrix with entries for observed or inferred edges. Node features can include capitalization, leverage, liquidity ratios, funding maturity, market-based risk measures, and balance-sheet exposure summaries. Edge features can include exposure amount, maturity, collateral, direction, and an indicator of the edge type. The specification does not require the graph to be complete; missing edges are treated as unobserved relations rather than absent economic linkages.

A central design choice is that not every plausible relation should be treated as an active propagation channel. Too dense a graph creates computational load and spurious amplification, while too sparse a graph misses indirect contagion. The framework therefore uses an adaptive edge kernel that combines economic intensity, temporal persistence, and a threshold rule. One defensible formulation is

$$A_ij, t = 1[w_ij, t \geq \tau_t] \cdot \exp(-\lambda d_ij, t)$$

where w_ij,t is a normalized linkage intensity between i and j, τ_t is a time-varying activation threshold, d_ij,t is a distance or decay term reflecting maturity, liquidity, or sector dissimilarity, and λ governs the rate at which weak or distant relations are discounted. The

indicator function activates an edge only when the linkage exceeds the threshold. This formulation preserves interpretable economic meaning: an exposure that is small, transient, or semantically remote contributes less to message passing than a large, persistent, and direct exposure. The threshold can be adjusted in validation to control graph density, and the decay term prevents the graph from collapsing all plausible financial relations into a single homogeneous channel.

2.2 Adaptive Temporal Dependence

Systemic risk is path dependent. A shock that occurs when liquidity is abundant may be absorbed; the same shock during a funding squeeze may propagate rapidly. The graph model must therefore represent both cross-sectional relations and temporal memory. A simple approach is to stack graph snapshots, but this creates a large, irregular computation and can confound legitimate market dynamics with changes in data coverage. The proposed framework instead maintains a temporal state for each node and updates it using current graph aggregation and previous hidden state. Neighborhoods are sampled adaptively from A_t, with sampling probabilities biased toward edges that have recently carried high risk, high exposure, or high residual correlation. This is not a claim that such edges are always more important; it is a computational allocation rule that allows the model to concentrate capacity on active transmission channels while retaining an exploration probability for low-weight or newly observed edges.

The adaptive component has three functions. It controls graph density by retaining a fixed or budgeted number of neighbors per node, it reduces variance in mini-batch training by emphasizing stable economic linkages, and it permits the model to react to regime changes without rebuilding a dense adjacency matrix at every time step. The resulting graph is still grounded in explicit financial relations because the sampling propensity is derived from exposure, persistence, and observed stress indicators rather than from an unrestricted attention mechanism alone.

2.3 Systemic Risk Target

The target of the framework is not only a node-level probability of distress but also a system-level loss distribution. For each node i and time t, the model produces a risk score r_i,t. This score may represent a transformed probability of distress, a predicted contribution to system loss, or a composite severity measure. A system risk measure aggregates node risks through weights and network amplification. The economic interpretation is that individual distress matters directly through its portfolio and liability weights and indirectly through the distress of connected entities. The system target must therefore be trained with an aggregation rule that represents both direct and network-mediated losses, rather than by averaging independent node labels.

This formulation implies a distinction between predictive accuracy at the node level and accuracy of the system-level ranking. A model may predict individual distress probabilities reasonably well while mis-ranking the institutions that would generate the largest cascade. The learning objective and evaluation protocol must therefore include system-level loss and rank-based diagnostics, not only standard classification metrics.

3. Adaptive GPU-Accelerated Graph Learning Architecture

3.1 Node, Edge, and Feature Construction

The architecture consumes a sequence of typed graph snapshots. Node features are standardized within rolling windows to avoid leakage from future periods. Edge features are represented in two forms. Continuous edge attributes enter a relation-specific projection, while edge type is encoded as a discrete channel. The graph is stored in compressed sparse row or column formats depending on whether aggregation is organized by source or destination. For a directed financial exposure from i to j, source-wise aggregation computes the outbound pressure exerted by i, while destination-wise aggregation computes the inbound vulnerability of j. The model maintains separate channels for these directions and combines them through learned gates.

Feature construction must reflect the distinction between stock and flow variables. Balance-sheet levels, such as capital and leverage, are stocks; payment flows, collateral calls, and trading volumes are flows. Mixing them without scaling can produce unstable gradients. The framework therefore uses separate normalization streams and includes missingness indicators where data are incomplete. This design is consistent with the broader observation that graph learning performance depends on preserving node, edge, and global context rather than relying on topology alone [14], though the specific economic feature set remains the responsibility of the risk modeler.

3.2 Temporal Message Passing

The core representation layer is a localized message-passing operator. At layer l, node states are updated by aggregating information from active neighbors and combining it with the node's own state. A normalized graph convolution form is

$$H_t\hat{}(l+1) = \sigma(\hat{A}_t\, H_t\hat{}(l)\, W_l + B_l)$$

where H_t^(l) is the node-state matrix at layer l, Â_t is a degree-normalized adjacency with self-loops, W_l is a learned weight matrix, B_l is a bias or feature projection, and σ is a pointwise nonlinearity. The normalization prevents high-degree nodes from dominating the aggregation merely because they have many connections. The self-loop retains the node's own risk profile, which is essential in finance because an institution's direct capital and liquidity position can dominate its short-run resilience. This layer is a temporal extension of graph convolution principles [8], and relation-specific attention weights can be used to distinguish edge types [9].

Stacking several message-passing layers yields higher-order neighborhood information. However, deeper propagation can oversmooth node states and make distinct institutions appear artificially similar. The framework therefore limits depth and uses residual connections and gated aggregation. The final graph representation is passed to a recurrent memory:

$$Z_t = \varphi(U\, Z_\{t-1\} + V\, H_t\hat{}(L) + b)$$

where Z_t is the temporal node state, H_t^(L) is the output of the final graph layer, U and V are learned transition matrices, b is a bias, and φ is a gated activation such as a sigmoid or hyperbolic tangent. The recurrent term allows a funding shock or liquidity deterioration to persist across snapshots. The graph term allows the state to incorporate new relational evidence. Inductive neighborhood aggregation is important because new institutions or newly observed edges can appear after training [10], and the original graph neural network formulation provides the conceptual basis for recursive state propagation over graph structure [11].

3.3 GPU Kernel Design and Sparse Execution

Systemic risk graphs are sparse but irregular. A dense matrix formulation is wasteful because most institution pairs have no direct exposure or common holding, and the active edge set can change across time. The computational workload is dominated by sparse neighborhood aggregation, feature projection, edge-wise attention, and repeated temporal updates. Accelerator-oriented graph convolution designs emphasize that performance is determined not only by arithmetic throughput but also by memory hierarchy, dataflow, and irregular access patterns [12]. The framework therefore treats GPU execution as a co-design problem. The adjacency is partitioned into tiles with nonzeros sorted by source or destination, and the aggregation kernel uses segment reductions so that a warp or block accumulates contributions for a node without serializing over all neighbors. Edge features and node features are stored in structures that permit coalesced reads for the dominant access pattern.

Kernel fusion reduces launch overhead and avoids writing intermediate neighbor messages to global memory. For example, edge-weight multiplication, nonlinear gating, and partial aggregation can be fused when the layer does not require a global normalization that depends on all node outputs. Mixed precision can be used for message aggregation and matrix multiplication when numerical sensitivity is controlled, while risk-sensitive outputs and loss accumulation remain in higher precision. The architecture also uses localized spectral or polynomial filters where a fixed receptive field is sufficient, because such filters can be expressed as sparse operations and avoid explicit eigendecomposition at each snapshot [13]. This choice is not a claim that spectral filtering always dominates message passing; it reflects a trade-off between expressive relational attention and predictable sparse execution.

The required reference position [12] is used here to support the general point that GCN accelerator design depends on dataflow and memory organization. The present framework does not assume that a particular accelerator architecture is universally optimal. Instead, it requires the execution plan to match graph density, edge-type cardinality, and temporal batch size. When the graph is very sparse, a gather-based neighbor aggregation may be preferable; when nodes have many neighbors and edge features are shared, a tiled sparse matrix multiplication may be more efficient. The model exposes these choices as configuration parameters rather than hard-coding one kernel schedule.

3.4 Adaptive Sampling and Workload Balance

Adaptive neighborhood sampling is necessary for two reasons. First, high-degree institutions can have thousands of plausible relations, and exact aggregation over all of them may be economically redundant. Second, temporal mini-batches must be balanced across nodes with very different activity levels. The sampling distribution for node i at time t is proportional to a score that combines normalized exposure intensity, recent edge activity, edge-type priority, and a small exploration term. The exploration term ensures that newly formed or previously low-weight relations can enter the computation. The sampler returns a fixed maximum number of neighbors per node, which bounds memory use and makes kernel shapes more regular.

Workload balance is handled by partitioning edges so that each processing unit receives a similar number of nonzeros rather than a similar number of nodes. This distinction matters because a naive node-based partition can assign one block a set of high-degree institutions and another block many low-degree institutions, producing poor utilization. The framework also uses a persistent-kernel strategy for repeated temporal snapshots so that graph metadata and projection matrices remain resident on the device. Batch construction is overlapped with computation through asynchronous transfers, and stale-neighborhood caching is permitted

only when the cache can be validated against a timestamp or a graph-change counter. These implementation details are part of the methodological contribution because they determine whether the adaptive graph learning model can be used on realistic multi-snapshot risk assessments.

4. Risk Estimation, Uncertainty, and Calibration

4.1 Temporal Covariance and Exposure Features

Graph relations based on exposures are not sufficient by themselves. Market co-movement and volatility dependence provide complementary information about common shocks and correlated liquidity pressure. The framework therefore augments graph features with a dynamic covariance representation. A practical decomposition is

$$\Sigma_t = D_t R_t D_t$$

where Σ_t is the conditional covariance matrix of relevant market returns or risk factors, D_t is a diagonal matrix of conditional volatilities, and R_t is a dynamic correlation matrix. Dynamic conditional correlation models provide a well-established way to represent time-varying correlations [18], while shrinkage estimators improve the conditioning of large covariance matrices when the number of assets is large relative to the sample window [19]. The covariance matrix is not used as a dense input to the graph network. Instead, it is summarized into edge features for selected pairs, node-level factor exposures, and a correlation-cluster embedding. This avoids quadratic memory growth and keeps the graph sparse.

The covariance features serve a specific purpose: they allow the model to distinguish two institutions that share a graph edge from two institutions that are statistically coupled even without a direct contractual exposure. The former suggests a direct contagion channel; the latter suggests common-factor sensitivity or correlated funding pressure. In risk assessment, conflating these channels can lead to over-attribution of spillovers to direct exposures or under-attribution of common vulnerabilities.

4.2 Composite Risk Objective

Node-level risk scores are produced from the temporal node state and selected economic features. A generic form is

$$r_i, t = \psi(z_i, t, c_i, t, s_i, t)$$

where r_i,t is the risk score or distress probability for node i, z_i,t is the learned temporal state, c_i,t is a vector of capital, liquidity, and leverage features, and s_i,t is a stress indicator derived from market or funding conditions. The function ψ can be implemented as a small neural network or a calibrated generalized linear head. The separation between z_i,t and c_i,t is important because it allows the model to learn relational effects while retaining direct financial covariates. The stress indicator s_i,t allows the mapping from state to risk to vary across regimes.

The system-level target aggregates node risks through economic weights and network amplification:

$$S_t = \sum_i \pi_i r_i, t + \kappa \sum_{i,j} A_i j, tr_i, tr_j, t$$

where S_t is the system risk score, π_i is the direct system weight of node i, A_ij,t is the active normalized adjacency, and κ controls the strength of pairwise network amplification. The first term captures direct loss or importance; the second captures joint stress on connected pairs. The multiplicative form reflects the intuition that two connected institutions under simultaneous stress can generate a larger system effect than the sum of their individual risks. The parameter κ is not an empirical constant to be asserted; it is a policy or model parameter that should be calibrated against historical stress episodes or set through scenario analysis. The expression is an analytical aggregation rule, not a claim that realized losses have been measured in this paper.

4.3 Predictive Uncertainty

Systemic risk decisions require uncertainty estimates. Two forms are relevant. Aleatoric uncertainty arises from intrinsic randomness in market outcomes, while epistemic uncertainty arises from limited data, model misspecification, or unfamiliar graph structures. The framework can represent epistemic uncertainty through approximate Bayesian inference or Monte Carlo dropout [16], and heteroscedastic output heads can model input-dependent noise [17]. Calibration is assessed on out-of-sample periods and stress windows, with reliability diagrams and proper scoring rules for probability outputs. Rank stability is also important: a model that changes the top-risk institution after small input perturbations is unsuitable for supervisory use even if its average calibration is acceptable.

The risk score should be accompanied by a decomposition into direct balance-sheet vulnerability, graph-mediated pressure, and uncertainty. This decomposition permits a risk officer to ask whether an institution is ranked highly because of its own leverage, because of its position in the exposure network, or because the model is extrapolating into a sparse region of the feature space. Such decompositions do not establish causality, but they make the model's reasoning inspectable and support challenge by domain experts.

5. Methodological Framework and Analytical Assessment

5.1 Learning Procedure

Training proceeds by temporal walk-forward windows. At each step, the model receives graph snapshots up to time t and is trained to predict risk and system loss over a subsequent horizon. The loss combines predictive risk loss, temporal smoothness, calibration, and regularization:

$$L = L_risk + \alpha L_smooth + \beta L_cal + \gamma \|\Theta\|_2\text{^}2$$

where L_risk is the primary risk or system-loss objective, L_smooth penalizes unstable temporal changes in node states that are not supported by feature changes, L_cal measures calibration error for probabilistic outputs, Θ denotes the model parameters, and α, β, and γ control the trade-offs. The smoothness term is not intended to suppress genuine regime changes. It penalizes high-frequency movement in predicted risk when the underlying graph and features are stable. The calibration term aligns predicted probabilities with observed distress frequencies, which is essential because systemic risk scores are often used comparatively and in threshold-based escalation rules.

The training procedure requires careful separation of information sets. Covariance features, graph edges, and node features must be computed using only information available at or before the prediction time. If balance-sheet data are reported with a lag, the lag must be represented explicitly. If edge data are inferred from market co-movement, the estimation

window must not overlap the evaluation horizon. These are standard concerns in financial forecasting, but they are especially consequential in graph learning because neighborhood aggregation can propagate future information if temporal boundaries are not enforced.

5.2 Propagation Operator and Stress Semantics

The propagation operator combines two distinct notions of connectedness. The graph adjacency encodes economic linkages, while the learned message function encodes how stress is transformed as it moves across a linkage. This separation is valuable because an exposure edge may transmit losses in one direction but transmit funding pressure in another. The framework therefore uses relation-specific transformations and directional channels. In analytical terms, the effective propagation strength from i to j is not A_ij,t alone but the composition of the edge weight, the learned gate, and the receiving node's vulnerability. This composition helps prevent the model from treating all edges as equivalent carriers of risk.

The risk propagation semantics also impose a boundary condition. Message passing should not be allowed to amplify risk without limit across repeated layers. The normalization in the graph convolution and the gated recurrent update both act as stabilizers. In addition, the model can enforce a monotonicity constraint on selected exposure features, so that an increase in leverage or a decrease in liquidity cannot reduce predicted distress when all other inputs are held fixed. Such constraints are not universally appropriate, but they can improve credibility in supervised risk applications where the direction of a first-order effect is well established.

5.3 Complexity and GPU Scaling

The computational complexity of exact message passing is linear in the number of active edges and quadratic in the hidden dimension for feature projections. With adaptive sampling, the per-batch aggregation cost is bounded by the number of sampled neighbors rather than the full graph degree. If N_active is the number of active nodes, d is the hidden dimension, and E_batch is the number of sampled edges in a temporal batch, the dominant cost is order O(E_batch d + N_active d^2) per layer, ignoring lower-order activation and normalization costs. This expression follows from sparse aggregation plus dense feature transformation. It explains why GPU acceleration must optimize both sparse gather/scatter operations and dense matrix multiplication; improving only one side leaves the other as a bottleneck.

Temporal batching introduces additional memory pressure because several graph snapshots may be required to compute a recurrence. The framework uses truncated backpropagation through time and caches only the snapshots needed for the current window. When graph topology changes slowly, adjacency metadata can be reused across adjacent snapshots; when it changes rapidly, the sampler must rebuild neighborhood lists. The trade-off is between memory reuse and staleness. A timestamped cache with an invalidation rule provides a practical compromise.

5.4 Evaluation Logic and Error Modes

An evaluation protocol for this framework must be defined before any empirical claim is made. It should include walk-forward prediction of node distress and system loss, comparison against econometric connectedness measures, static network models, and graph-learning baselines without temporal memory or adaptive sampling. Metrics should include ranking quality for the most systemically important nodes, calibration of distress probabilities, stability of rankings across adjacent periods, and performance on stress windows rather than only on average periods. The protocol should also include ablations of edge types, covariance

features, adaptive sampling, and GPU-specific approximations such as mixed precision or truncated neighborhoods. These ablations test whether the computational design changes risk conclusions, not merely runtime.

Several domain-specific error modes require attention. False negatives near a crisis are more costly than false positives in ordinary periods, but an evaluation that overweights crisis periods can produce a model that is unusable outside them. Graph leakage can occur when future exposures are used to construct current edges. Oversmoothing can make distinct institutions appear similar and suppress the tail risk of idiosyncratic institutions. Adaptive sampling can introduce bias if it systematically ignores weak but structurally important edges. Calibration can fail when the feature distribution shifts even if ranking remains stable. Explainability methods can produce plausible attributions that do not correspond to causal propagation. The framework addresses these risks through temporal separation, stability regularization, uncertainty reporting, and explicit comparison of sampled and full-neighborhood predictions on a validation subsample.

6. Technical Discussion

6.1 Comparison with Econometric and Network Models

Econometric connectedness measures provide interpretable summaries of spillovers and are well suited to low-dimensional or aggregated data [1], [2]. Network models provide explicit contagion mechanisms and can represent cascades that correlation measures miss [3], [4]. The proposed graph-learning framework is not a replacement for these approaches. Its advantage is representational: it can combine multiple edge types, nonlinear feature interactions, and temporal memory in a single predictive system. Its disadvantage is that learned propagation is less transparent than a fixed accounting identity or a DebtRank rule. The appropriate comparison is therefore not whether graph learning always dominates, but whether it improves out-of-sample system-level risk ranking and calibration after controlling for the information available to simpler models.

The framework also differs from static network risk models in its treatment of time. A static adjacency may be adequate for a stress test with a fixed horizon, but it cannot capture the formation and dissolution of linkages around funding stress. The temporal state Z_t allows the model to represent persistence and regime dependence. However, temporal flexibility raises the risk of overfitting to transient market patterns. Regularization, walk-forward validation, and parsimonious edge construction are necessary counterweights.

6.2 Adaptivity, Stability, and Explainability Trade-offs

Adaptive sampling and temporal attention increase model capacity, but they also increase variance. A highly adaptive model may revise its ranking of systemic institutions after small changes in edge weights, which is problematic for supervisory communication. The framework therefore treats stability as a first-class objective. Smoothness penalties, ensembling across sampling seeds, and rank-stability diagnostics can reduce excessive sensitivity. Explainability methods such as GNNExplainer provide local explanations by identifying influential subgraphs [21], while axiomatic attribution methods provide feature-level attributions with desirable theoretical properties [22]. These tools should be used to support expert review, not to assert causal identification. A risk analyst should be able to distinguish an institution that is central because of direct exposures from one that is central because of common holdings or correlated market factors.

There is also a trade-off between graph density and interpretability. Dense learned graphs can capture subtle dependence but are difficult to audit and computationally expensive. The adaptive edge kernel in Section 2.1 provides a middle ground by restricting propagation to linkages that exceed an economic or statistical threshold. The threshold itself should be documented and stress-tested, because it defines which channels the model can see.

6.3 Implementation Conditions and Supervisory Use

Supervisory use imposes requirements beyond predictive performance. The model must have a documented information set, reproducible preprocessing, versioned graph construction, and a clear update cadence. Data gaps must be represented rather than silently imputed in a way that creates artificial edges. The system should log graph density, sampling rates, kernel configurations, and uncertainty summaries for each assessment. Human review should focus on changes in the top-risk set, large attribution shifts, and periods in which predictive uncertainty is high. The framework is designed to support such review, but it does not automate supervisory judgment.

Computational efficiency is necessary but not sufficient. GPU acceleration enables repeated temporal message passing and scenario analysis, but it can also encourage excessive model complexity. The design should be parsimonious where economic relationships are well understood and expressive only where heterogeneous relational data justify additional capacity. In this sense, the contribution of the framework is organizational as much as algorithmic: it aligns graph representation, risk semantics, uncertainty, and execution so that each component supports the others.

7. Conclusion

This paper has developed an adaptive GPU-accelerated graph learning framework for systemic risk assessment in interconnected financial markets. The framework represents financial markets as dynamic, typed, and sparse graphs, uses temporal message passing to combine direct exposures with higher-order neighborhood information, and aggregates node-level risk into a system-level loss measure that reflects network amplification. It incorporates covariance-based exposure features, a composite risk objective, calibration and uncertainty reporting, and explainability tools. The GPU execution design addresses the specific computational bottlenecks of sparse graph aggregation, adaptive neighborhood sampling, temporal recurrence, and workload imbalance.

The central conclusion is that graph learning for systemic risk should be treated as a coupled representational and computational problem. A model that ignores the economic semantics of edges may learn spurious connectivity, while a model that ignores GPU execution constraints may be too slow for repeated stress assessment or may rely on approximations that change the risk ranking. The proposed framework provides a disciplined way to connect these concerns. It also clarifies the conditions under which graph learning is most likely to add value: when multiple relational channels matter, when temporal dependence is strong, and when system-level ranking and calibration are evaluated rather than only average node-level accuracy.

The framework does not claim that empirical superiority has been established. Its contribution is methodological: a precise formulation of the graph, propagation, risk aggregation, learning objective, uncertainty logic, computational complexity, and evaluation criteria required for credible use. Future analytical work can refine the edge-kernel calibration, formalize monotonicity constraints, and compare alternative aggregation rules. Empirical

validation, when undertaken, should follow the walk-forward protocol and stress-window diagnostics specified here, so that performance claims are tied to a defensible information set and to system-level risk conclusions rather than to isolated prediction metrics.

References

1. Billio, M., Getmansky, M., Lo, A. W., & Pelizzon, L. (2012). Econometric measures of connectedness and systemic risk in the finance and insurance sectors. Journal of Financial Economics, 104(3), 535–559.
2. Diebold, F. X., & Yılmaz, K. (2014). On the network topology of variance decompositions: Measuring the connectedness of financial firms. Journal of Econometrics, 182(1), 119–134.
3. Acemoglu, D., Ozdaglar, A., & Tahbaz-Salehi, A. (2015). Systemic risk and stability in financial networks. American Economic Review, 105(2), 564–608.
4. Battiston, S., Puliga, M., Kaushik, R., Tasca, P., & Caldarelli, G. (2012). DebtRank: Too central to fail? Financial networks, the FED and systemic risk. Scientific Reports, 2, 541.
5. Haldane, A. G., & May, R. M. (2011). Systemic risk in banking ecosystems. Nature, 469(7330), 351–355.
6. Caccioli, F., Barucca, P., & Kobayashi, T. (2018). Network models of financial systemic risk: A review. Journal of Computational Social Science, 1(1), 81–114.
7. Barabási, A.-L., & Albert, R. (1999). Emergence of scaling in random networks. Science, 286(5439), 509–512.
8. Kipf, T. N., & Welling, M. (2017). Semi-supervised classification with graph convolutional networks. In International Conference on Learning Representations.
9. Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., & Bengio, Y. (2018). Graph attention networks. In International Conference on Learning Representations.
10. Hamilton, W. L., Ying, Z., & Leskovec, J. (2017). Inductive representation learning on large graphs. In Advances in Neural Information Processing Systems (Vol. 30, pp. 1024–1034).
11. Scarselli, F., Gori, M., Tsoi, A. C., Hagenbuchner, M., & Monfardini, G. (2009). The graph neural network model. IEEE Transactions on Neural Networks, 20(1), 61–80.
12. Xie, X., Peng, H., Hasan, A., Huang, S., Zhao, J., Fang, H., ... & Ding, C. (2023, October). Accel-gcn: High-performance gpu accelerator design for graph convolution networks. In 2023 IEEE/ACM International Conference on Computer Aided Design (ICCAD) (pp. 01-09). IEEE.
13. Defferrard, M., Bresson, X., & Vandergheynst, P. (2016). Convolutional neural networks on graphs with fast localized spectral filtering. In Advances in Neural Information Processing Systems (Vol. 29, pp. 3844–3852).
14. Zhou, J., Cui, G., Hu, S., Zhang, Z., Yang, C., Liu, Z., Wang, L., Li, C., & Sun, M. (2020). Graph neural networks: A review of methods and applications. AI Open, 1, 57–81.
15. Xu, K., Hu, W., Leskovec, J., & Jegelka, S. (2019). How powerful are graph neural networks? In International Conference on Learning Representations.

16. Gal, Y., & Ghahramani, Z. (2016). Dropout as a Bayesian approximation: Representing model uncertainty in deep learning. In Proceedings of the 33rd International Conference on Machine Learning (pp. 1050–1059).

17. Kendall, A., & Gal, Y. (2017). What uncertainties do we need in Bayesian deep learning for computer vision? In Advances in Neural Information Processing Systems (Vol. 30, pp. 5574–5584).

18. Engle, R. (2002). Dynamic conditional correlation: A simple class of multivariate generalized autoregressive conditional heteroskedasticity models. Journal of Business & Economic Statistics, 20(3), 339–350.

19. Ledoit, O., & Wolf, M. (2004). A well-conditioned estimator for large-dimensional covariance matrices. Journal of Multivariate Analysis, 88(2), 365–411.

20. Battiston, S., Caldarelli, G., May, R. M., Roukny, T., & Stiglitz, J. E. (2016). The price of complexity in financial networks. Proceedings of the National Academy of Sciences, 113(36), 10031–10036.

21. Ying, Z., Bourgeois, D., You, J., Zitnik, M., & Leskovec, J. (2019). GNNExplainer: Generating explanations for graph neural networks. In Advances in Neural Information Processing Systems (Vol. 32, pp. 9240–9251).

22. Sundararajan, M., Taly, A., & Yan, Q. (2017). Axiomatic attribution for deep networks. In Proceedings of the 34th International Conference on Machine Learning (pp. 3319–3328).