Software Engineering and Artificial Intelligence Systems

Volume: 02 Issue: 01 (2026)

www.seais.org/

Cross-Attack Generalization of Temporal Graph Neural Networks under Strategic Adversarial Drift

Daniel Rios[1]

[1]Computer Science and Engineering, College of Engineering, University of Michigan, Ann Arbor, MI, USA

Keyword: temporal graph neural networks, adversarial robustness, fraud detection, dynamic graphs, unforeseen attacks, invariance.

A B S T R A C T

Temporal graph neural networks are the standard detector architecture for fraud in continuous-time transaction and interaction streams, and the most effective recent defences train them against a simulated adversary so that the decision boundary anticipates strategic evasion. Those defences are trained and evaluated under a single threat model, so what they establish is robustness along the axis the defender has already chosen. This paper measures what happens when the adversary changes tactic rather than intensity. We place four evasion primitives—edge injection, attribute manipulation, event pacing and identity substitution—inside one perturbation space over the event stream, instantiate six attack families in it, and score every defence on a cross-attack robustness matrix in which three families are visible during training and three are held out. A first-order analysis shows that adversarial training constrains the loss gradient only inside the span of the families it sees, leaving out-of-family sensitivity free whenever the operator blocks are misaligned. Experiments on two real dynamic graphs and a synthetic anti-money-laundering stream match the prediction: single-family adversarial training gains up to +49.1 F1 points on its own family while surrendering up to 59.1 points on another, and the average fraction of its in-family gain that reaches other families is 0.02. We then introduce CAGE, which couples simplex-sampled composite perturbations with an attack-agnostic embedding-smoothness term and a block-isotropic gradient penalty. CAGE raises worst-case F1 by +18.8 points over an undefended detector and by +10.0 points over the strongest alternative defence on each stream, with a transfer ratio of 1.47.

INTRODUCTION

Financial crime is now a graph problem. Card networks, correspondent-banking rails, crypto-asset exchanges and peer-to-peer rating platforms all emit a continuous stream of timestamped interactions, and the signatures that distinguish a laundering ring from an unusual customer live in the topology and the timing of that stream rather than in any single record. Continuous-time dynamic graphs (CTDGs) are the natural representation, and temporal graph neural networks (TGNNs) that maintain per-entity memory states and update them at every event have become the default detector family [1]–[5].

What separates this domain from ordinary supervised learning is that the data-generating process reads the model. A deployed decision boundary changes the payoff of every evasion strategy available to the criminal, so the future distribution is a function of the current detector—the setting studied as performative prediction [6] and strategic classification [7]. The drift that results is directed rather than stochastic, and periodic retraining on freshly labelled data is always one step behind it. The strongest response in the recent literature is to bring the adversary inside the training loop. Fan et al. [8] formalise fraud detection as a continuous Stackelberg game in which a learned attack generator proposes future structural perturbations and the detector optimises against them, and they report substantial gains in F1 under adaptive attack together with a sharp reduction in catastrophic forgetting, at a training overhead small enough for production use. That work establishes that anticipating the adversary is both tractable at industrial throughput and worth its cost. It also raises a question that the standard evaluation protocol cannot answer. In essentially every adversarial-training study on graphs, the perturbations used at training time and the perturbations used at test time are drawn from the same family and differ only in budget. Robustness is therefore certified along the one axis the defender selected in advance. A criminal organisation faces no such constraint. It picks whichever evasion is cheapest given the deployed system, and the ranking of costs moves as the system moves: when injected endorsement edges stop paying, the same ring re-paces its transfers to exploit memory staleness, splits value across freshly registered mule accounts, or simply adjusts transfer amounts. Strategic drift is movement between families at least as much as movement within one. This paper studies that movement directly. We ask whether robustness acquired against one evasion family transfers to families the defender never simulated, and we answer with a measurement protocol, a first-order analysis and a training method. The measurement rests on a single observation: the evasion strategies available against a TGNN all act on the same object, the event stream, and can be written as perturbations of four separable primitives—which interactions exist, what attributes they carry, when they occur, and which identity carries them. Collecting these into one unified perturbation space makes attack families comparable, makes composition well defined, and makes the seen/held-out split explicit. We instantiate six families: three gradient-driven white-box families exposed to the defences during training, and three held out—a black-box attack transferred from an independently trained surrogate, an identity-substitution attack that activates a block no defence has ever seen, and a compound adversary that optimises across all blocks at once.

The analysis explains why transfer should not be expected for free. To first order the loss increase a family can extract is the norm of the loss gradient projected onto the subspace that family spans. Adversarial training on family A penalises that projection for A and says nothing about the orthogonal complement, so the residual available to a second family B is governed by a misalignment coefficient that equals one whenever B reaches outside the span of A. Robustness transfers only through whatever coupling the shared representation happens to provide, which is exactly what our matrices measure. The method follows from the analysis. If single-family training controls one direction and union training controls a finite set of directions, a defence that generalises must either cover the convex geometry of the space rather than its vertices, or shrink sensitivity in a way that is not tied to any family. CAGE does both: it samples composite perturbations from a Dirichlet simplex over the primitive blocks so that training sees mixtures rather than pure families, it penalises the movement of node embeddings under random, non-adversarial stream edits, and it equalises the per-block sensitivity of the loss so that no block remains a cheap evasion route. Our contributions are as follows.

1) A unified perturbation space for evasion against CTDG detectors, with realisability constraints proper to an append-only ledger, and a taxonomy of six attack families instantiated inside it and split into seen and held-out groups.

2) An evaluation protocol built on the cross-attack robustness matrix, together with three summary quantities: the cross-attack generalisation gap, the cross-attack transfer ratio, and worst-case robustness across families.

3) A first-order result identifying subspace misalignment as the quantity that limits transfer, and a corollary that bounds what union adversarial training can achieve outside the families it enumerates.
4) CAGE, a training procedure combining simplex-sampled composite adversarial training, an attack-agnostic embedding-smoothness term and a block-isotropic gradient penalty.
5) An empirical study on two real dynamic graphs and one synthetic anti-money-laundering stream across three seeds, which quantifies the negative transfer produced by single-family adversarial training and shows that CAGE removes most of it.

II. RELATED WORK

A. Learning on continuous-time dynamic graphs

JODIE [1] and DyRep [2] introduced coupled recurrent updates for interacting entities; TGAT [3] added a functional time encoding with temporal attention; and TGN [4] unified memory modules, message functions and temporal aggregation into the template that most production systems follow. Later work questioned how much architectural complexity is needed—GraphMixer [5] matches attention-based encoders with fixed time windows and a multilayer perceptron—and studied the expressive power of memory-based encoders [9]. Evaluation practice for dynamic graphs has itself been revisited, with negative-sampling strategy shown to dominate reported numbers [10]. This literature optimises accuracy on benign streams; robustness is not part of its design objective.

B. Adversarial attacks and defences on graphs

Static graph attacks are mature. Nettack [11] perturbs structure and features against a target node, metattack [12] poisons training through meta-gradients, and reinforcement learning [13] and first-order topology optimisation [14] provide alternative search procedures. Defences clean or constrain the graph—low-rank filtering [15], learned structure refinement [16], neighbour reweighting [17], robust aggregation at scale [18]—and a systematic re-evaluation found that many of them lose most of their advantage under adaptive attack [19]; surveys cover the space in full [20]. The dynamic setting is younger. Sharma et al. [21] attack discrete-time models under a perturbation constraint that preserves the rate of change of the graph, Lee et al. [22] give attacks and defences for link prediction on continuous-time graphs, and generative models have been used to synthesise anomalous edge distributions for training [23]. Fan et al. [8] close the loop by optimising the detector against a learned generator inside a bilevel objective and by retaining topologically significant history in a motif-based replay buffer. All of these evaluate under the threat model they train against, which is precisely the convention this paper puts under test.

C. Robustness beyond one threat model

In vision, the limits of single-threat robustness are established. Adversarial training against an L-infinity ball [24], [25] yields little robustness to L1 or L2 attacks, and training on a union of balls costs accuracy on each [26], [27]. Kang et al. [28] made the point a measurement problem by scoring defences against distortions never seen in training, and perceptual threat models [29] widen the ball rather than enumerate it; reliable evaluation requires diverse attack ensembles [30]. Our seen/held-out protocol is the graph-stream analogue of that agenda, with the additional structure that the primitives are semantically interpretable operations on a ledger rather than norms on a pixel grid.

D. Invariance, smoothness and adversarial drift

Methods that seek predictors stable across environments are the natural source of attack-agnostic regularisers: invariant risk minimisation [31], gradient matching across domains [32] and agreement-based masking [33] all penalise solutions that work for one environment only. Smoothness-based training is the other source, from virtual adversarial training [34] to the accuracy-robustness trade-off analysed by TRADES [35] and to certified smoothing [36]. On the drift side, continual graph learning mitigates forgetting with experience replay [37] and

generative replay [38] when the distribution moves for non-adversarial reasons. CAGE borrows the environment view from the first group and the sensitivity view from the second, and applies both to perturbation families rather than to data domains.

III. PROBLEM SETTING

A. Detector

A continuous-time dynamic graph is a stream of events E = {e_1, ..., e_M} ordered by time, with e_i = (u_i, v_i, t_i, x_i): a source entity, a destination entity, a timestamp and an attribute vector x_i in R^d holding transferred value, channel and causal summaries of the recent past. The detector assigns to every event a probability that it is illicit,

$$\hat{y}_i = f_\theta(e_i \mid H_i) = \sigma(g_\theta([\, z_{u_i}, z_{v_i}, x_i, \varphi(\Delta t_i)\,])) \quad (1)$$

where H_i is the history strictly preceding t_i, φ is a learned time encoding and z_u is the embedding of entity u at t_i^-. Entity state is carried by a memory vector s_u updated by a gated recurrent cell on every incident event,

$$m_u = \psi_\theta(s_u, s_v, x_i, \varphi(\Delta t_i)), \quad s_u \leftarrow \mathrm{GRU}(m_u, s_u) \quad (2)$$

and the embedding concatenates memory, a neighbourhood summary and the encoded recency, z_u = tanh(W [s_u ; n_u ; φ(Δt_u)]). This is the memory-plus-message template of TGN [4]; the analysis and the method below are stated for it but use no property beyond differentiability of f_θ in its inputs.

B. Threat model

The adversary controls one endpoint of the events it wishes to hide—the rated account in a web of trust, the originating account in a transfer—and wants those events scored as benign. Four constraints make the setting concrete. First, the ledger is append-only: historical events cannot be deleted and past timestamps cannot be rewritten, so every structural perturbation is an addition carrying a timestamp no earlier than the current window. Second, counterparties are not free variables: a new edge must terminate at an account that would plausibly interact with the attacker, which we model by drawing candidate endpoints from a pool of accounts with established histories. Third, every block carries a budget, and attribute values stay inside the box observed on the training stream. Fourth, the adversary optimises the detector's loss, using either the detector's own gradients (white box) or those of an independently trained surrogate (black box).

C. Unified perturbation space

Write Π_δ(E) for the stream obtained by applying a perturbation δ to E. We decompose δ into four blocks,

$$\delta = (\delta_S, \delta_X, \delta_T, \delta_I) \in \Delta = \Delta_S \times \Delta_X \times \Delta_T \times \Delta_I \quad (3)$$

with the following semantics. The structural block $\delta_S \in [0,1]^C$ selects, per target event, which of C candidate endorsement edges the adversary buys; each selected edge adds one message to the controlled entity's memory before the decision, and the block budget is $\|\delta_S\|_1 \le \kappa$. The attribute block δ_X perturbs the observable attributes of the controlled event inside an L-infinity ball of radius ε_X intersected with the training box. The timing block δ_T shifts the log inter-arrival gaps of the controlled entity by at most ε_T, modelling an operator who delays a transfer or pads it with filler activity. The identity block $\delta_I \in [0, \alpha]$ routes the flow through a freshly created account whose state is a convex combination of the controlled memory and a prototype built from accounts with clean histories, $s \leftarrow (1-\delta_I)\, s + \delta_I\, \hat{s}$. Composition is well defined because the blocks act on disjoint coordinates of the event representation, and a family is simply a subset of blocks together with its budgets. Fig. 1 summarises the space.

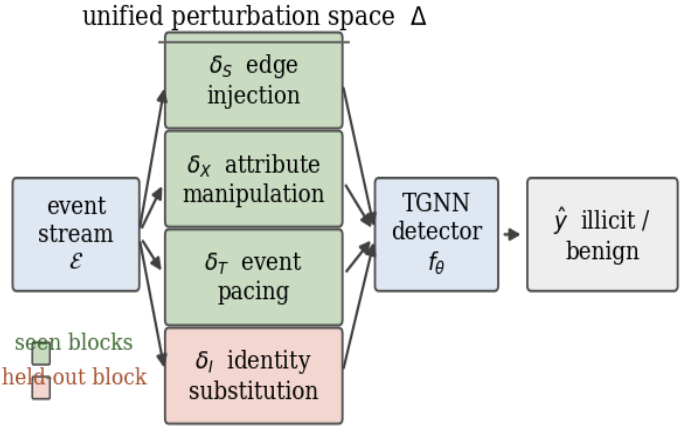


Figure 1The unified perturbation space. Four primitive blocks act on the event stream before it reaches the detector. Three are exposed to the defences during training; identity substitution is held out.

D. Attack families

Table II lists the six families. A1, A3 and A4 are white-box projected-gradient attacks restricted to one block each, and are the families the adversarially trained defences see. The held-out group is designed so that each member differs from the seen group along a different axis. A2 changes the information available to the attacker: perturbations are computed on a surrogate detector trained independently on the same stream and then applied to the victim, which is the realistic situation for an adversary without white-box access. A5 changes the operator: it activates the identity block, which no defence has trained on. A6 changes the geometry: it optimises all four blocks jointly under the same per-block budgets, so the attack direction lies in the interior of the space rather than on an axis.

E. Cross-attack evaluation

Let $R_F(\theta)$ denote the positive-class F1 of detector θ measured on the test stream perturbed by family F, and let S and U be the seen and held-out family sets. The cross-attack robustness matrix collects R_F for every defence and every family. We summarise it with three quantities. The cross-attack generalisation gap

$$G(\theta) = (1/|S|) \Sigma_{F\in S} R_F(\theta) - (1/|U|) \Sigma_{F\in U} R_F(\theta) \quad (4)$$

measures how much of a detector's apparent robustness is confined to the families it was built for. The cross-attack transfer ratio compares gains against an undefended reference θ_0,

$$T(\theta) = [\Sigma_{F\in U} \Delta R_F] / [\Sigma_{F\in S} \Delta R_F], \quad \Delta R_F = R_F(\theta) - R_F(\theta_0) \quad (5)$$

so that $T = 1$ means robustness transfers in full and $T < 0$ means a defence buys in-family robustness by giving up out-of-family robustness. Finally, worst-case robustness $W(\theta) = \min_F R_F(\theta)$ is the quantity an operator actually experiences, because the adversary chooses the family.

IV. WHY IN-FAMILY ROBUSTNESS NEED NOT TRANSFER

Fix θ and an event, and let $\ell(\delta)$ be the loss as a function of the perturbation, with $g = \nabla_\delta \ell(0)$. Let a family F occupy the coordinate subspace spanned by an orthonormal basis U_F, with feasible set contained in an ε-ball of that subspace, and write $P_F = U_F U_F^T$.

Proposition 1 (first-order family reach). For a perturbation set $B_F = \{\delta \in \text{span}(U_F) : \|\delta\|_2 \le \varepsilon\}$, the loss increase achievable by F satisfies $\max_{\delta\in B_F} \ell(\delta) - \ell(0) = \varepsilon \|P_F g\| + O(\varepsilon^2)$. If adversarial training on family A drives $\|P_A g\| \le \xi$, then for any other family B

$$\|P_B g\| \le \xi + \kappa_{BA} \|g\|, \quad \kappa_{BA} = \|P_B (I - P_A)\|_2 \quad (6)$$

and the bound is tight; in particular $\kappa_{BA} = 1$ whenever $\text{span}(U_B)$ is not contained in $\text{span}(U_A)$.

Proof. The first claim is Cauchy–Schwarz applied to the linearisation $\ell(\delta) = \ell(0) + g^{T}\delta + O(\|\delta\|^2)$, with the maximiser $\delta = \varepsilon P_F g / \|P_F g\|$. For the second, split $g = P_A g + (I - P_A) g$ and apply the triangle inequality together with $\|P_B P_A g\| \le \|P_A g\|$. The operator norm $\|P_B (I - P_A)\|_2$ is the cosine of the smallest principal angle between span(U_B) and span$(U_A)^R$, which equals one exactly when span(U_B) contains a direction orthogonal to span(U_A). Choosing g along that direction attains the bound. ■ Two consequences matter here. First, because our blocks act on disjoint coordinates, $\kappa = 1$ for every pair of distinct blocks: a defence that drives the structural gradient to zero has placed no constraint at all on the attribute, timing or identity gradient, and any transfer observed in practice is an artefact of the shared encoder, not of the training objective. Second, the same argument applied to a union defence trained on blocks $A_1, \ldots, A_k$ bounds only $\|P_{\{A_1 \cup \ldots \cup A_k\}} g\|$, so a held-out block remains unconstrained no matter how many families are enumerated. Reducing $\|g\|$ itself is the only mechanism in (6) that acts on families outside the enumerated set, which is why CAGE carries an explicit sensitivity term alongside its adversarial term.

The analysis is first order and therefore describes the local geometry that a gradient-based adversary exploits at the operating point; it does not by itself predict the size of the empirical gap, which depends on how strongly the encoder couples the blocks. Measuring that coupling is the purpose of Section VII.

V. CAGE

CAGE (cross-attack generalised training) keeps the adversarial term that makes anticipatory defences effective and adds two terms that are not tied to any family.

A. Simplex-sampled composite perturbations

Union adversarial training evaluates the loss under each enumerated family and backpropagates through the maximiser, so the perturbations it ever sees lie on the axes of Δ. Held-out families rarely do. At each step CAGE instead draws mixing weights from a Dirichlet distribution over the seen blocks, $\lambda \sim \mathrm{Dir}(\alpha\, 1)$, and runs projected gradient ascent under the reweighted budgets

$$\varepsilon_b(\lambda) = |B|\, \lambda_b\, \bar{\varepsilon}_b, \quad b \in \{S, X, T\} \qquad (7)$$

so the total budget is preserved in expectation while its allocation varies. With $\alpha < 1$ the draw is sparse and the axes are still visited; with $\alpha \ge 1$ the interior dominates. We use $\alpha = 0.7$ and two ascent steps, which costs less than one union step because only one maximisation is run.

B. Attack-agnostic embedding smoothness

The second term asks the encoder to place nearby streams at nearby embeddings, with "nearby" defined by random, non-adversarial edits rather than by an attack. For a batch we draw ξ consisting of random endorsement weights, Gaussian attribute noise and random non-negative pacing shifts, and penalise embedding movement

$$L_{EMS}(\theta) = E_\xi \left[\| z_u(\xi) - z_u \|^2 + \| z_v(\xi) - z_v \|^2 \right] \qquad (8)$$

Because ξ is sampled rather than optimised, this term shrinks the sensitivity of the representation in expectation over the whole space, including the coordinates of blocks that no attack in training activates. It is the term that reaches the identity block, whose effect is by construction a displacement of the memory state.

C. Block-isotropic gradient penalty

Proposition 1 says the reach of a family is set by the norm of the loss gradient in its block. If training leaves one block far more sensitive than the others, that block becomes the cheapest evasion route. The third term equalises the profile by penalising the squared block-wise directional derivatives

$$R_{BIG}(\theta) = \Sigma_b \left(\partial \ell / \partial \delta_b \right)^2 \approx \Sigma_b \left[(\ell(\eta\, u_b) - \ell(0)) / \eta \right]^2 \qquad (9)$$

where u_b is a random unit direction confined to block b and η is a small step. The right-hand side is an unbiased finite-difference estimator of the directional derivative up to $O(\eta)$, so the penalty needs one extra forward pass per block and no second-order differentiation. Sign

constraints of the physical operators are respected by drawing u_b from the feasible cone, for instance non-negative endorsement weights.

D. Objective and cost

Writing L_cls for the class-weighted cross-entropy on a batch and δ for the composite perturbation of Section V-A, the training objective is

$$L = L_cls(\theta) + L_cls(\theta, \delta) + \rho\, L_EMS(\theta) + \gamma\, R_BIG(\theta) \quad (10)$$

Each step runs one clean forward-backward pass, two ascent steps for δ, one adversarial pass, one perturbed pass for L_EMS and $|B| + 1$ forward passes for R_BIG, giving a constant-factor overhead over standard training with no Hessian or implicit-gradient computation. Measured end to end this is 3.23× the cost of a single-family adversarial scheme and 1.58× the cost of union adversarial training on our streams. Algorithm 1 gives the step.

Algorithm 1 CAGE training step

Input: batch B of events, parameters θ, memories S, budgets $\bar{\varepsilon}$, weights ρ, γ
1: refresh memories from pending messages; assemble endorsement candidates
2: $\lambda \sim$ Dir(α 1) over blocks {S, X, T}; set $\varepsilon_b = |B|\, \lambda_b\, \bar{\varepsilon}_b$
3: $\delta \leftarrow$ two steps of projected gradient ascent on $\ell(\theta, \cdot)$ under $\varepsilon(\lambda)$
4: draw ξ (random weights, attribute noise, pacing shifts) and evaluate L_EMS
5: for each block b: draw u_b, evaluate $\ell(\eta\ u_b)$, accumulate R_BIG
6: $\theta \leftarrow$ Adam step on L of (10)
7: commit the clean batch to memory and queue its messages

VI. EXPERIMENTAL SETUP

A. Data

Table I reports the three streams.

BTC-OTC is the Bitcoin-OTC web of trust [39], in which members rate counterparties from −10 to +10 after trading. We order the ratings by timestamp and label an event positive when the rating is negative, i.e. when a member reports a counterparty as untrustworthy. The rating of a past event is visible to the memory module as a message attribute but never to the classifier for the current event, so an entity's reputation has to be carried by its memory state. This is the setting in which buying endorsements is the canonical evasion.

Tennis-UO is the Twitter mention graph collected around the 2017 US Open [40], with accounts as entities and mentions as timestamped interactions. An event is positive when the mentioned account is about to undergo an attention surge, defined as a mention volume in the next interval exceeding three times its trailing five-interval mean. The task is a behavioural analogue of coordinated-promotion detection and is included because its dynamics are bursty in a way transaction data is not.

AML-Sim is a synthetic transaction stream with preferential-attachment counterparty selection, log-normal amounts and three planted laundering typologies: fan-out smurfing just below a reporting threshold, layering chains with decaying amounts, and cyclical round-tripping. It supplies the one thing the public streams cannot, a ground truth in which the illicit subgraphs are known by construction and the label depends on topology, amount structure and timing simultaneously. Public labelled financial graphs such as the Elliptic Bitcoin set [42] are the usual reference point for this task; the synthetic stream complements them by making the generating typologies explicit.

All streams are split chronologically into 70% training, 15% validation and 15% test. Attribute standardisation and every auxiliary statistic—entity degrees, the reputable candidate pool, the clean-account prototype—are computed on the training portion only.

B. Detector and defences

The detector is the memory-based encoder of Section III-A with 40-dimensional memory, an 8-dimensional time encoding, 40-dimensional embeddings and a two-layer head, trained with

Adam at $3\times10^{-4} \times 10$ for three epochs over the training stream in batches of 200 events, with class weighting capped at 12. Seven training schemes are compared: undefended empirical risk minimisation; random-noise adversarial training, which perturbs each batch with unoptimised noise; three single-family adversarial trainings using two-step projected gradient ascent restricted to the structural, attribute and timing blocks; union adversarial training, which takes the maximum loss over the three single-block attacks [26]; and CAGE with $\rho = 1.0$, $\gamma = 0.3$ and $\alpha = 0.7$. All adversarially trained schemes see exactly the same three blocks and the same budgets.

C. Protocol

Every scheme is trained from three seeds. The decision threshold is selected on the unperturbed validation stream by maximising F1 and then frozen, so that a defence cannot recover robustness by re-tuning its operating point after seeing the attack. Attacks are applied to the test window only, targeting positive events, which is the evasion setting: the adversary wants illicit activity scored as benign and has no interest in the rest of the stream. During training, perturbations are applied to the whole batch rather than to positives only, so that the presence of a perturbation carries no label information. Budgets are $\kappa = 4$ endorsement edges, $\varepsilon_X = 0.4$ standard deviations, $\varepsilon_T = 3.0$ log-time units and $\alpha = 0.6$ for the identity block. The surrogate used by A2 is an independent detector trained with a different seed. We report positive-class F1; AUC follows the same ordering.

Table 1STREAM STATISTICS

Stream	Events	Entities	Attr.	Positive	Test events
BTC-OTC	35,592	5,881	9	10.0%	5,339
Tennis-UO	43,387	1,000	10	24.1%	6,509
AML-Sim	34,961	2,500	10	3.1%	5,245

Table 2ATTACK FAMILIES IN THE UNIFIED PERTURBATION SPACE

Family	Blocks	Knowledge	Budget	Description
A1 Struct	S	white box	$\kappa = 4$ edges	buys endorsement edges from established accounts
A3 Attr	X	white box	$\varepsilon_X = 0.4$	rewrites observable attributes of the event
A4 Time	T	white box	$\varepsilon_T = 3.0$	delays or pads activity to reshape recency
A2 Transfer	S, X	black box	as A1, A3	optimised on an independent surrogate detector

A5 Identity	I	no gradient	$\alpha = 0.6$	routes the flow through a fresh mule account
A6 Compound	all	white box	per-block	optimises every block jointly

Top group: families visible to the adversarially trained defences. Bottom group: held out.

VII. RESULTS

A. The vulnerability profile of an undefended detector

Table III gives the full matrix and Table IV its summary. An undefended detector is vulnerable on every stream: BTC-OTC falls from 0.554 clean to 0.360 under comp; Tennis-UO falls from 0.740 clean to 0.002 under comp; AML-Sim falls from 0.856 clean to 0.609 under comp. The vulnerability is strongly anisotropic. The spread between the mildest and the harshest family is 0.18 on BTC-OTC, 0.66 on Tennis-UO, 0.25 on AML-Sim F1 points, and the ordering of families is not the same on any two streams—the attribute block dominates on BTC-OTC, structural and timing manipulation dominate on Tennis-UO, and no single block dominates on AML-Sim, where the label depends on amount structure as much as on topology. This is the first practical consequence of the problem we study: a defender who hardens the block that matters on one deployment has no reason to expect the same block to matter on the next. Compound optimisation is the strongest attack everywhere, so the blocks combine rather than compete.

B. Single-family adversarial training buys robustness on credit

Every single-family scheme improves the family it trains on, and the improvements are large. The biggest is +49.1 F1 points, obtained by AT-time against time on Tennis-UO. Read only along that column, each of these defences looks like a success. The rest of the row tells the other half of the story. Averaged over the five families it does not train on, AT-attr on Tennis-UO loses 18.6 points relative to an undefended detector, and its single largest degradation is 59.1 points against ident. Attribute-only training is the clearest case of this reallocation: it is the only scheme that is worse than no defence at all on 3 of the three streams once the average over the other families is taken.

The transfer ratio of (5) makes the accounting explicit. Across the nine single-family configurations the mean ratio is 0.02, so almost none of the in-family gain reaches the families the defence did not simulate, and two configurations have a negative ratio, meaning the gain is paid for out of robustness elsewhere. Union training over the three seen blocks behaves better on the seen group, as the literature on multiple perturbation models would predict [26], [27], but Proposition 1 bounds what it can do outside that group: with three blocks enumerated and one held out, its held-out mean improves by +5.0, +0.2, +2.4 points on the three streams, and on the identity block specifically it moves by −1.5, −53.6, −1.8 points—close to the zero the misalignment argument predicts when $\kappa = 1$.

C. CAGE

CAGE attains the best worst-case robustness on all three streams. Against the undefended detector the improvement in W is +14.2 on BTC-OTC, +27.5 on Tennis-UO, +14.6 on AML-Sim points; against the strongest alternative defence on each stream it is +6.6 on BTC-OTC, +20.5 on Tennis-UO, +2.8 on AML-Sim points. Tennis-UO is the extreme case: every other scheme, including union training, is driven to a worst-case F1 of at most 0.072 by the family it handles least well, while CAGE retains 0.276. An operator who assumes the adversary picks the cheapest family is choosing between a detector that still works and one that does not.

The gain is concentrated where the analysis says it should be. The held-out mean rises by +8.3, +15.4, +6.8 points over undefended training, the generalisation gap falls from 0.111 averaged over the single-family schemes to 0.081, and the transfer ratio rises from 0.02 to 1.47: CAGE is the only scheme whose out-of-family gain is of the same order as its in-family gain. Fig. 2 plots the seen and held-out means side by side, and Fig. 3 renders the matrix as deviations from the undefended baseline, where the red band across the attribute-training row is the negative transfer of Section VII-B.

The identity block is the cleanest test of the argument, because no defence trains on it and no attack in training activates it. Single-family and union schemes move it by −1.5, −53.6, −1.8 points, which is the behaviour (6) predicts for an unconstrained block. CAGE moves it by −0.6, −30.1, −0.9 points, and the ablation attributes that to the smoothness term rather than to the adversarial term.

D. Ablation

Fig. 4 and the underlying numbers isolate each component. Removing simplex sampling in favour of union maximisation over the same three blocks costs 21.1 points on the compound family and 19.5 on the worst case, by far the largest effect of the three: the compound direction lies in the interior of the space that mixtures cover and axis attacks do not. Removing the embedding-smoothness term costs 15.9 points on the identity family, the largest identity-specific effect, which is consistent with its role as the only component that constrains blocks never activated in training; on AML-Sim it also costs clean accuracy, because the term doubles as a regulariser on a stream with few positives. Removing the block-isotropic penalty has the mildest effect on the held-out mean but still costs 11.7 points of worst-case F1, and the loss is concentrated on whichever seen block was cheapest to begin with—on BTC-OTC the attribute family falls from 0.488 to 0.442 when sensitivity is no longer equalised. No component accounts for the result alone.

E. Budget and hyperparameters

Fig. 5 traces the compound adversary as the per-block budgets are scaled from half to twice their nominal value. CAGE is the strongest defence at every budget on both streams, and the margin widens rather than closes as the adversary gets more room: its lead over the undefended detector grows from +7.0 and +4.3 points at half budget to +16.6 and +13.3 points at double budget on BTC-OTC and AML-Sim respectively. The ordering among the baselines is less stable—union training sits below undefended training on AML-Sim at every budget—which is a further symptom of optimising against an enumerated set whose members are not the binding constraint. Fig. 6 sweeps the two weights of the objective on BTC-OTC. Worst-case robustness rises monotonically in the smoothness weight ρ over the range examined, from 0.477 to 0.488, and clean F1 rises with it; behaviour in the gradient-penalty weight γ is comparatively flat. The two roles are visible in that asymmetry: ρ acts on the total sensitivity that (6) leaves free, while γ redistributes sensitivity between blocks without changing its total.

F. Cost and utility

Clean F1 moves by +1.5 on BTC-OTC, −12.8 on Tennis-UO, +0.1 on AML-Sim points relative to undefended training. On the two streams where the undefended detector is already close to its ceiling the change is within noise; on Tennis-UO, where every family is devastating and the undefended worst case is 0.002, CAGE trades clean accuracy for a worst case of 0.276. That exchange is the one an operator facing an adaptive adversary would choose, and it is smaller than the exchange the single-family schemes make implicitly: on the same stream AT-attr keeps 0.709 clean F1 while its mean over the families it does not cover falls 18.6 points below the undefended model. Table V reports training cost. CAGE runs at 1.58× the cost of union adversarial training and 3.23× the cost of a single-family scheme, the overhead coming from the extra smoothness pass and the $|B| + 1$ forward passes of the finite-difference penalty. No Hessian, implicit gradient or inner-loop unrolling is required, and inference is unchanged, since all three added terms exist only at training time.

Table 3CROSS-ATTACK ROBUSTNESS MATRIX (POSITIVE-CLASS F1, MEAN OF THREE SEEDS)

Defence	**Clean**	**A1 str**	**A3 att**	**A4 tim**	**A2 trf**	**A5 idn**	**A6 cmp**
BTC-OTC							
ERM	0.554	0.531	0.470	0.538	0.416	**0.541**	0.360
Noise-AT	0.545	0.528	0.457	0.533	0.436	0.537	0.388
AT-A1 struct	0.546	0.545	0.449	0.532	0.466	0.538	0.398
AT-A3 attr	**0.570**	0.378	**0.535**	0.571	0.364	0.523	0.323
AT-A4 time	0.551	0.492	0.473	**0.610**	0.399	0.535	0.444
Union-AT	0.561	0.534	0.526	0.564	0.504	0.527	0.437
CAGE (ours)	0.569	**0.569**	0.525	0.590	**0.528**	0.536	**0.502**
Tennis-UO							
ERM	0.740	0.256	0.467	0.262	0.052	**0.665**	0.002
Noise-AT	**0.743**	0.383	0.528	0.322	0.212	0.588	0.001
AT-A1 struct	0.695	**0.680**	0.348	0.258	0.341	0.600	0.001
AT-A3 attr	0.709	0.003	**0.680**	0.166	0.066	0.073	0.000
AT-A4 time	0.731	0.296	0.543	**0.753**	0.072	0.659	0.114
Union-AT	0.697	0.635	0.627	0.114	**0.595**	0.129	0.000
CAGE (ours)	0.612	0.584	0.535	0.540	0.540	0.363	**0.276**
AML-Sim							
ERM	0.856	0.854	0.754	0.823	0.758	0.830	0.609
Noise-AT	0.863	**0.861**	0.793	0.837	0.798	0.821	0.629
AT-A1 struct	0.865	0.859	0.795	0.846	0.803	0.823	0.662
AT-A3 attr	0.845	0.828	0.758	0.818	0.748	0.812	0.620
AT-A4 time	**0.865**	0.859	0.765	**0.870**	0.762	**0.848**	0.726
Union-AT	0.845	0.837	0.796	0.830	0.793	0.812	0.665

CAGE (ours)	0.857	0.852	**0.821**	0.858	**0.824**	0.821	**0.755**

Best value per column and stream in bold. A1/A3/A4 are seen by the adversarially trained defences; A2/A5/A6 are held out.

Table 4SUMMARY ROBUSTNESS STATISTICS

Defence	Clean	Own	ΔOwn	ΔOther	G	T	Worst W
BTC-OTC							
ERM	0.554	0.513	--	--	0.074	--	0.360
Noise-AT	0.545	0.506	−0.7	+1.4	0.053	--	0.388
AT-A1 struct	0.546	0.545	+1.4	+1.1	0.041	0.82	0.398
AT-A3 attr	0.570	0.535	+6.5	−4.5	0.091	-0.70	0.323
AT-A4 time	0.551	0.610	+7.3	+0.5	0.066	0.07	0.399
Union-AT	0.561	0.541	+2.8	+5.0	0.052	1.76	0.437
CAGE (ours)	0.569	0.561	+4.8	+8.3	0.039	1.71	**0.502**
Tennis-UO							
ERM	0.740	0.328	--	--	0.089	--	0.002
Noise-AT	0.743	0.411	+8.3	+2.8	0.144	0.33	0.001
AT-A1 struct	0.695	0.680	+42.4	+2.0	0.114	0.05	0.001
AT-A3 attr	0.709	0.680	+21.3	−18.6	0.236	-0.87	0.000
AT-A4 time	0.731	0.753	+49.1	+4.8	0.249	0.10	0.072
Union-AT	0.697	0.459	+13.0	+0.2	0.217	0.01	0.000
CAGE (ours)	0.612	0.553	+22.4	+15.4	0.160	0.69	**0.276**
AML-Sim							
ERM	0.856	0.810	--	--	0.078	--	0.609
Noise-AT	0.863	0.831	+2.0	+1.7	0.081	0.84	0.629
AT-A1 struct	0.865	0.859	+0.5	+3.1	0.071	--	0.662
AT-A3 attr	0.845	0.758	+0.4	−1.0	0.074	--	0.620

AT-A4 time	0.865	0.870	+4.8	+3.1	0.053	0.65	0.726
Union-AT	0.845	0.821	+1.1	+2.4	0.064	2.26	0.665
CAGE (ours)	0.857	0.844	+3.3	+6.8	0.044	2.02	**0.755**

Own is the family a single-family defence trains on, and the seen-family mean for the remaining schemes; ΔOwn and ΔOther are changes in F1 points against the undefended detector, Other being the mean over all families the scheme does not train on. G is the gap (4), T the transfer ratio (5) reported where ΔOwn exceeds one point, and W the minimum over the six families.

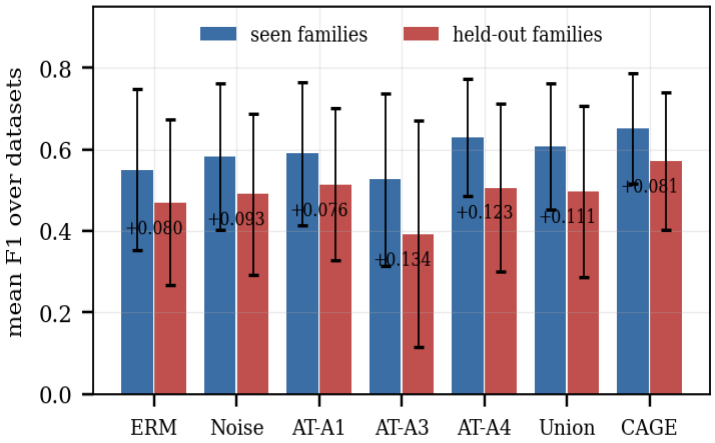


Figure 2. Mean F1 over the seen and held-out family groups, averaged over the three streams; error bars are the spread across streams and the annotation is the cross-attack generalisation gap.

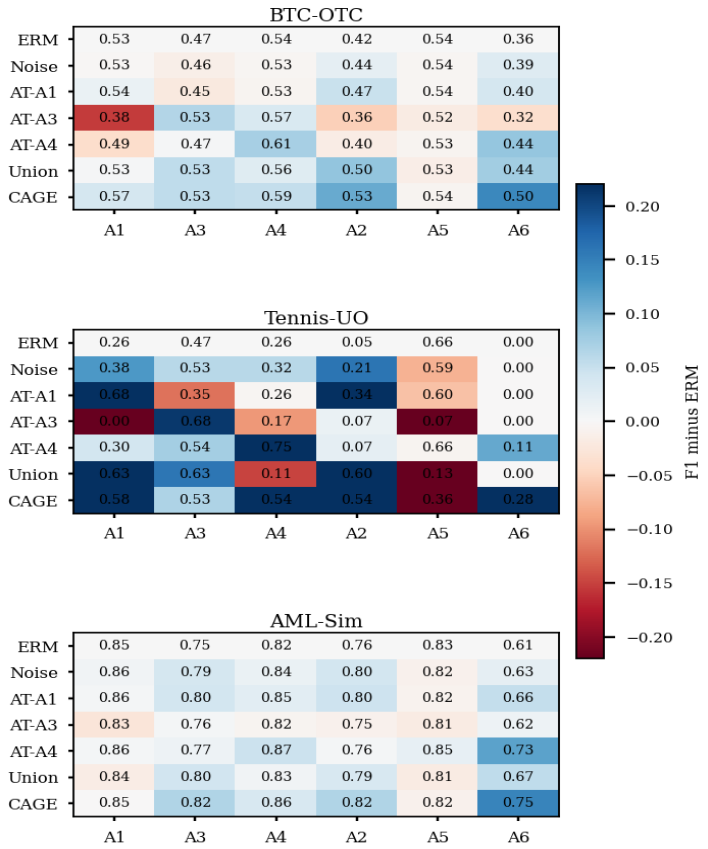


Figure 3Cross-attack robustness matrix expressed as the change in F1 relative to undefended training. Blue is improvement, red is degradation.

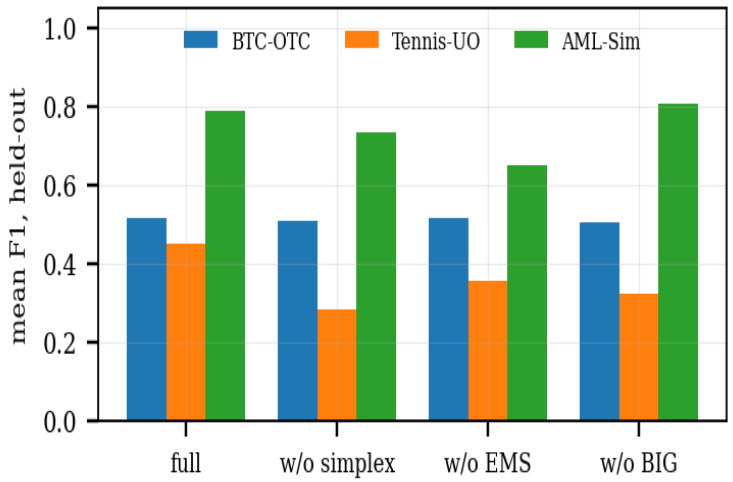


Figure 4 Ablation: mean F1 over the held-out families when one component of CAGE is removed.

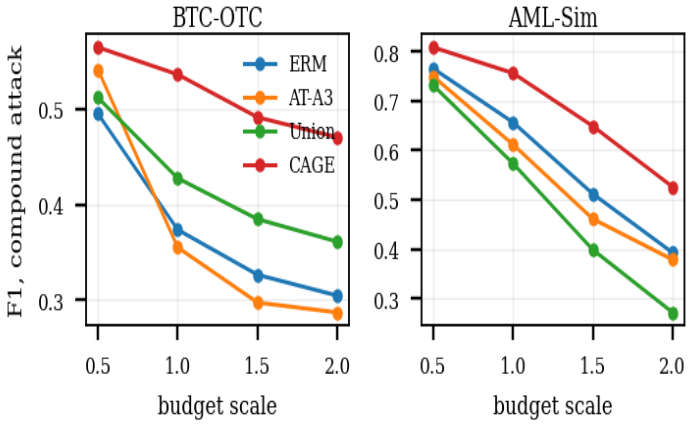


Figure 5Worst-case behaviour under the compound adversary as the per-block budgets are scaled.

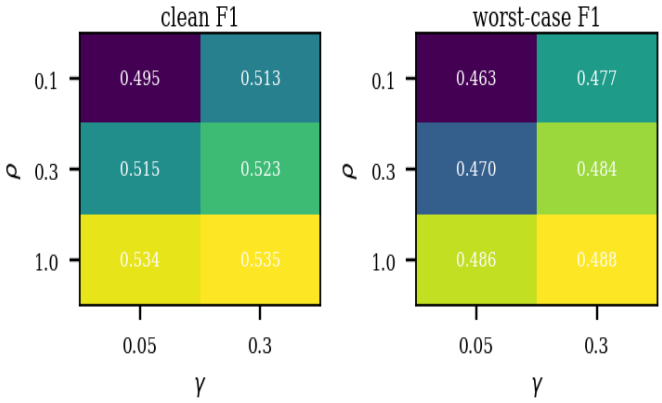


Figure 6Sensitivity of CAGE to the smoothness weight ρ and the gradient-penalty weight γ on BTC-OTC.

Table 5TRAINING COST PER RUN (SECONDS, THREE EPOCHS)

Defence	**BTC-OTC**	**Tennis-UO**	**AML-Sim**
ERM	2.7	2.9	4.4
Noise-AT	12.8	15.1	20.7
AT-A1 struct	19.0	24.8	34.3
AT-A3 attr	20.1	25.9	33.2

AT-A4 time	21.1	35.8	34.4
Union-AT	42.5	67.1	56.9
CAGE (ours)	63.9	110.8	90.0

VIII. CONCLUSION

Anticipating the adversary is the right instinct for fraud detection on dynamic graphs, and the recent Stackelberg formulations show how much can be gained by putting an attack generator inside the training loop. This paper adds the axis those formulations do not measure. By expressing edge injection, attribute manipulation, event pacing and identity substitution as blocks of one perturbation space, we can hold a defence to the standard an operator actually faces: performance against whichever family the adversary finds cheapest, including families the defender never simulated. Under that standard, single-family adversarial training is a reallocation of robustness rather than an increase in it, and the first-order account in Proposition 1 explains why—the training objective constrains the loss gradient only inside the span it enumerates. CAGE closes most of the gap by covering the interior of the space with simplex-sampled mixtures, by shrinking representation sensitivity with an attack-agnostic smoothness term that reaches blocks no attack activates, and by equalising sensitivity across blocks so that none of them stays cheap. The resulting detector has the best worst-case F1 on all three streams at a modest constant-factor training cost.

Two directions follow naturally. The first is to widen the space: coordinated account creation, multi-entity collusion and heterogeneous entity types all admit block representations and would extend the taxonomy without changing the protocol. The second is to replace the sampled smoothness term with a learned sampler over the whole space, which would make the attack-agnostic component adaptive while keeping it independent of any enumerated family.

REFERENCES

S. Kumar, X. Zhang, and J. Leskovec, “Predicting dynamic embedding trajectory in temporal interaction networks,” in Proc. 25th ACM SIGKDD Int. Conf. Knowledge Discovery & Data Mining, 2019, pp. 1269–1278.

R. Trivedi, M. Farajtabar, P. Biswal, and H. Zha, “DyRep: Learning representations over dynamic graphs,” in Proc. Int. Conf. Learning Representations, 2019.

D. Xu, C. Ruan, E. Korpeoglu, S. Kumar, and K. Achan, “Inductive representation learning on temporal graphs,” in Proc. Int. Conf. Learning Representations, 2020.

E. Rossi, B. Chamberlain, F. Frasca, D. Eynard, F. Monti, and M. Bronstein, “Temporal graph networks for deep learning on dynamic graphs,” arXiv preprint arXiv:2006.10637, 2020.

W. Cong, S. Zhang, J. Kang, B. Yuan, H. Wu, X. Zhou, H. Tong, and M. Mahdavi, “Do we really need complicated model architectures for temporal networks?” in Proc. Int. Conf. Learning Representations, 2023.

J. Perdomo, T. Zrnic, C. Mendler-Dünner, and M. Hardt, “Performative prediction,” in Proc. Int. Conf. Machine Learning, 2020, pp. 7599–7609.

M. Hardt, N. Megiddo, C. Papadimitriou, and M. Wootters, “Strategic classification,” in Proc. ACM Conf. Innovations in Theoretical Computer Science, 2016, pp. 111–122.

H. Fan, Y. Jiao, M. Wang, J. Chen, and S. Yang, “Fraud learns too: Continual graph learning under strategic adversarial drift in dynamic networks,” Scientific Reports, 2026, doi: 10.1038/s41598-026-60997-7.

A. Souza, D. Mesquita, S. Kaski, and V. Garg, “Provably expressive temporal graph networks,” in Advances in Neural Information Processing Systems, vol. 35, 2022, pp. 32257–32269.

F. Poursafaei, S. Huang, K. Pelrine, and R. Rabbany, “Towards better evaluation for dynamic link prediction,” in Advances in Neural Information Processing Systems, vol. 35, 2022.

D. Zügner, A. Akbarnejad, and S. Günnemann, "Adversarial attacks on neural networks for graph data," in Proc. 24th ACM SIGKDD Int. Conf. Knowledge Discovery & Data Mining, 2018, pp. 2847–2856.
D. Zügner and S. Günnemann, "Adversarial attacks on graph neural networks via meta learning," in Proc. Int. Conf. Learning Representations, 2019.
H. Dai, H. Li, T. Tian, X. Huang, L. Wang, J. Zhu, and L. Song, "Adversarial attack on graph structured data," in Proc. Int. Conf. Machine Learning, 2018, pp. 1115–1124.
N. Entezari, S. A. Al-Sayouri, A. Darvishzadeh, and E. E. Papalexakis, "All you need is low (rank): Defending against adversarial attacks on graphs," in Proc. 13th ACM Int. Conf. Web Search and Data Mining, 2020, pp. 169–177.
S. Geisler, T. Schmidt, H. Şirin, D. Zügner, A. Bojchevski, and S. Günnemann, "Robustness of graph neural networks at scale," in Advances in Neural Information Processing Systems, vol. 34, 2021.
F. Mujkanovic, S. Geisler, S. Günnemann, and A. Bojchevski, "Are defenses for graph neural networks robust?" in Advances in Neural Information Processing Systems, vol. 35, 2022, pp. 8954–8968.
K. Sharma, R. Trivedi, R. Sridhar, and S. Kumar, "Temporal dynamics-aware adversarial attacks on discrete-time dynamic graph models," in Proc. 29th ACM SIGKDD Conf. Knowledge Discovery and Data Mining, 2023, pp. 2023–2035.
D. Lee, J. Lee, and K. Shin, "Spear and shield: Adversarial attacks and defense methods for model-based link prediction on continuous-time dynamic graphs," in Proc. AAAI Conf. Artificial Intelligence, vol. 38, 2024, pp. 13374–13382.
I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," in Proc. Int. Conf. Learning Representations, 2015.
A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, "Towards deep learning models resistant to adversarial attacks," in Proc. Int. Conf. Learning Representations, 2018.
F. Tramèr and D. Boneh, "Adversarial training and robustness for multiple perturbations," in Advances in Neural Information Processing Systems, vol. 32, 2019.
P. Maini, E. Wong, and Z. Kolter, "Adversarial robustness against the union of multiple perturbation models," in Proc. Int. Conf. Machine Learning, 2020.
Fang, Q., & Liu, W. (2026). Neuro-symbolic pedagogical alignment (NSPA) for long-horizon classroom discourse analysis: Mitigating dialect bias via counterfactual preference optimization. Computers and Education: Artificial Intelligence, 100664.
Xing, S., Wang, Y., & Liu, W. (2025). Self-adapting CPU scheduling for mixed database workloads via hierarchical deep reinforcement learning. Symmetry, 17(7), 1109.
Ji, E., Wang, Y., Xing, S., & Jin, J. (2025). Hierarchical reinforcement learning for energy-efficient API traffic optimization in large-scale advertising systems. IEEE Access.
Wang, Y., & Xing, S. (2025). AI-driven CPU resource management in cloud operating systems. Journal of Computer and Communications, 13(06), 135-149.
Xing, S., & Wang, Y. (2025). Cross-modal attention networks for multi-modal anomaly detection in system software. IEEE Open Journal of the Computer Society.
D. Kang, Y. Sun, D. Hendrycks, T. Brown, and J. Steinhardt, "Testing robustness against unforeseen adversaries," arXiv preprint arXiv:1908.08016, 2019.
C. Laidlaw, S. Singla, and S. Feizi, "Perceptual adversarial robustness: Defense against unseen threat models," in Proc. Int. Conf. Learning Representations, 2021.
F. Croce and M. Hein, "Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks," in Proc. Int. Conf. Machine Learning, 2020.
M. Arjovsky, L. Bottou, I. Gulrajani, and D. Lopez-Paz, "Invariant risk minimization," arXiv preprint arXiv:1907.02893, 2019.
Y. Shi, J. Seely, P. H. S. Torr, N. Siddharth, A. Hannun, N. Usunier, and G. Synnaeve, "Gradient matching for domain generalization," in Proc. Int. Conf. Learning Representations, 2022.

Chen, T., Shang, W., & Guo, Z. (2024). Xsla: An SLA-aware heterogeneous XPU scheduling framework for edge AI inference. World Journal of Information Technology, 2(2), 48–59.

Zou, J., Qin, Y., & Rhee, M. (2024). Beyond model scale: A task-aware reliability benchmark for large language models in clinical decision support. Innovation and Technology Studies, 1(1), 39–47.

Fang, Q. (2024). Bias-aware large language models for equitable educational feedback and assessment. Trends in Social Sciences and Humanities Research, 2(1), 41–50.

Qin, Y., Zou, J., & Rhee, M. (2022). An AI-driven intelligent transaction execution framework for internet finance platforms: Cost-sensitive real-time integrity interdiction. Eurasia Journal of Science and Technology, 4(2), 36–44.

Wang, Y., & Zhang, C. (2023). AERIS: Adaptive executor scoped resource management for privacy preserving data processing in Apache Spark. Eurasia Journal of Science and Technology, 5(2), 25-33.

Zhao, W., & Wang, B. (2022). CAPA: A prediction-driven autoscaling framework for SLO-aware cloud-native machine learning inference. Journal of Computer Science and Electrical Engineering, 4(1), 8-16.

Guo, Z., Chen, T., & Shang, W. (2022). Class-conditional intermediate-domain adversarial adaptation for cross-device acoustic scene classification. Journal of Computer Science and Electrical Engineering, 4(1), 26-35

Shen, A. (2023). Compute-efficient scaling of large sequential recommendation models under dynamic user behavior. World Journal of Information Technology, 1(1), 62–71.

Fang, Q. (2024). True-EWS: Trustworthy early warning with equalized conformal guarantees for equitable student support in U.S. education. Educational Research and Human Development, 1(2), 44–54.

Shen, A., & Huang, S. (2022). A systems-aware framework for intelligent large-scale recommendation models. Journal of Computer Science and Electrical Engineering, 4(1), 36–44.