KDD 2026 August 9–13, 2026 Jeju, Korea

KAIST

Detecting Change, Not Just State:

Change-Aware Graph Learning for Evolving Fraud and Abuse

Chanyoung Park

Associate Professor
Industrial and Systems Engineering
Graduate School of AI
KAIST
cy.park@kaist.ac.kr

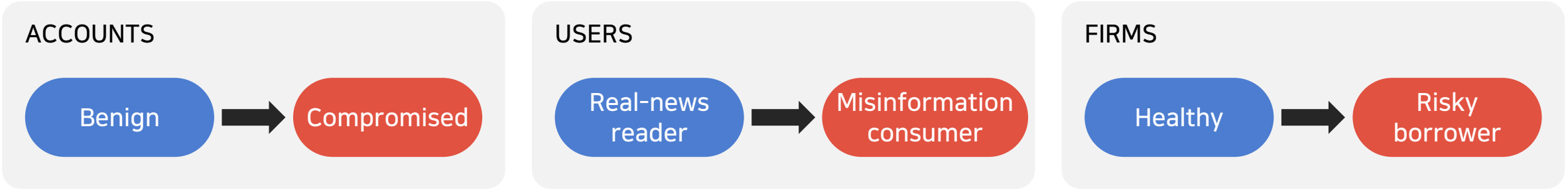
1st Workshop on AI for Fraud and Abuse (AI-FA) @ KDD 2026 · Jeju, Korea · August 10, 2026

Outline

- **Motivation** — fraud and abuse as moving targets
- **Chasing label shifters in dynamic graphs**
  - Why entities whose labels change are the ones that matter
  - Why all dynamic GNNs miss them
  - How to fix it
- **Lessons for AI for fraud & abuse**

Fraud and Abuse Are Moving Targets

- Fraudsters, abusers, and at-risk users **do not stand still** — they change state over time



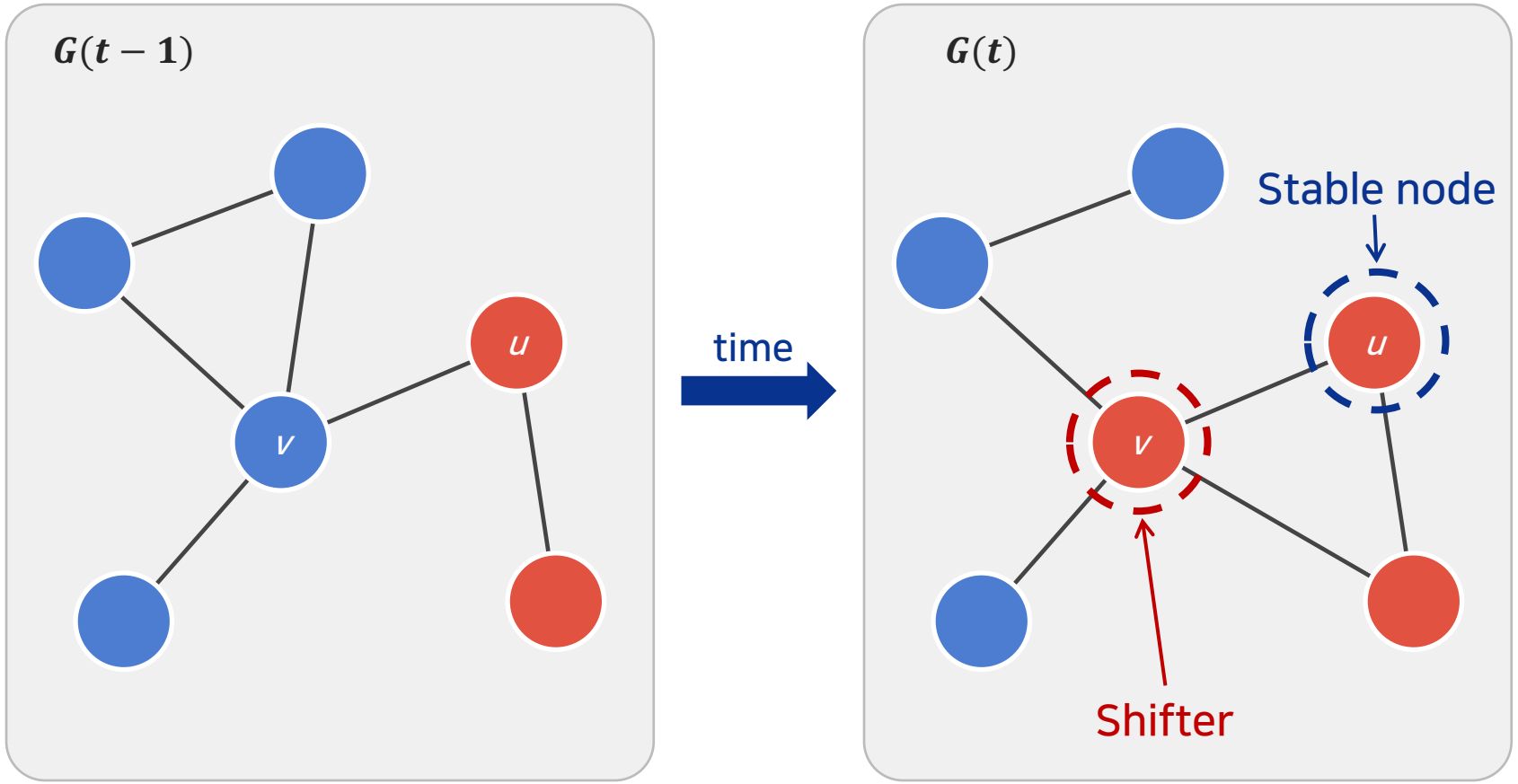
- Yet most detection pipelines assume **one fixed label per entity**
- The blind spot this talk focuses on

> **Models fail on entities whose labels change**
>
> When labels change, predictions tend to remain stuck in the past

Chasing Label Shifters

A Change-Aware Framework for Dynamic Graph Node Classification

Shifters vs. Stable Nodes



Class A Class B

Shifter

- label changes between snapshots
  - $y_v(t) \neq y_v(t-1)$ (node v: A → B)
- ex) A researcher changing fields, a user drifting into fake news, a firm sliding into a risky credit state

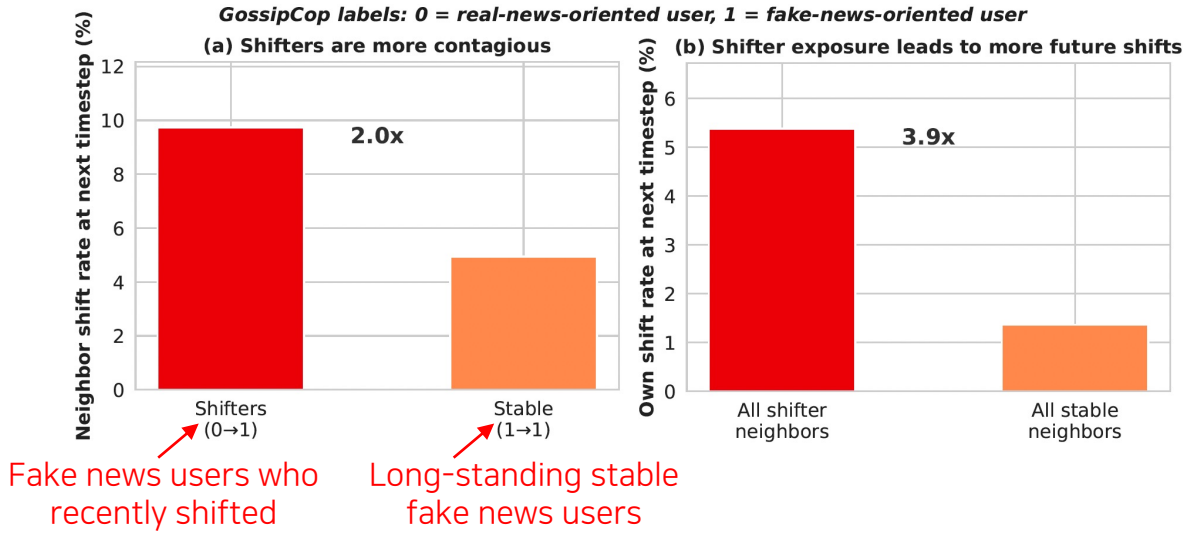
Stable node

- label persists across snapshots:
  - $y_v(t) = y_v(t-1)$ (node u: B → B)

Dynamic GNNs model evolving structure and features
— But how well do they classify the nodes whose labels evolve?

Why Shifters Matter: Misinformation Propagation

- **GossipCop**: users linked by co-consuming news; labeled fake-news consumer if >50% of weekly consumption is fabricated



- (Left) Neighbors of a **recent shifter** (0→1) shift at **2.0×** the rate of neighbors of a stable fake-news user (1→1) → Shifters are the contagious ones
- (Right) Users **surrounded by recent shifters** are **3.9×** more likely to shift than those surrounded by stable fake-news users → Shifts tend to propagate through the user network

Recent shifters are more influential than established fake-news users
→ Detecting them early enables timely intervention

Three Benchmarks with Naturally Shifting Labels

- Existing dynamic-graph benchmarks lack node features or meaningful label change

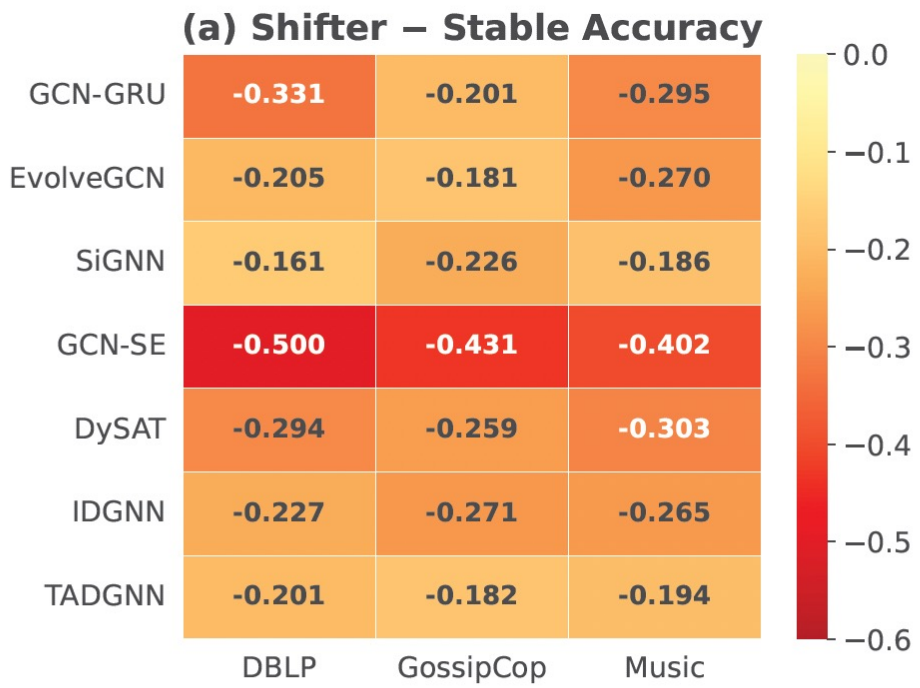
→ we construct three from real-world behavior

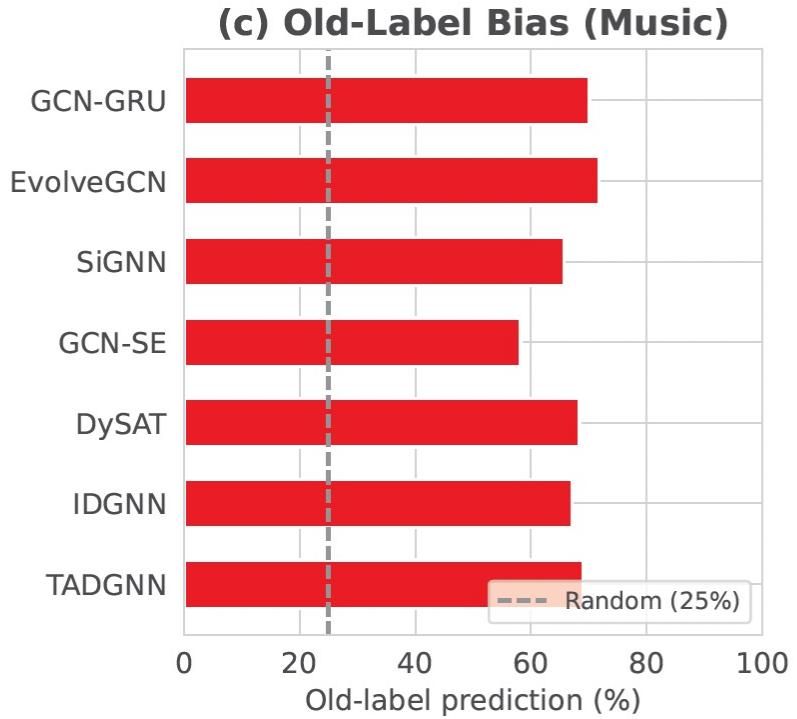
- Labels are **derived from observed behavior** and shift naturally — not synthetically injected

Dataset	Domain	# Nodes	# Edges	Snapshots	Classes	Label meaning	Shifter %
DBLP	Academic	34,717 (researcher)	397,312 (co-author)	14 (yearly)	2	Research area	2.62%
GossipCop	Social media	6,091 (news consumer)	200,390 (co-consume)	37 (weekly)	2	>50% fake-news consumption	4.06%
Music	Music streaming	878 (listener)	100,457 (co-listen)	24 (monthly)	5	Dominant genre	18.77%

What happens when we run the SOTA dynamic GNNs on these benchmarks?

A Universal Failure Mode: The Shifter Gap



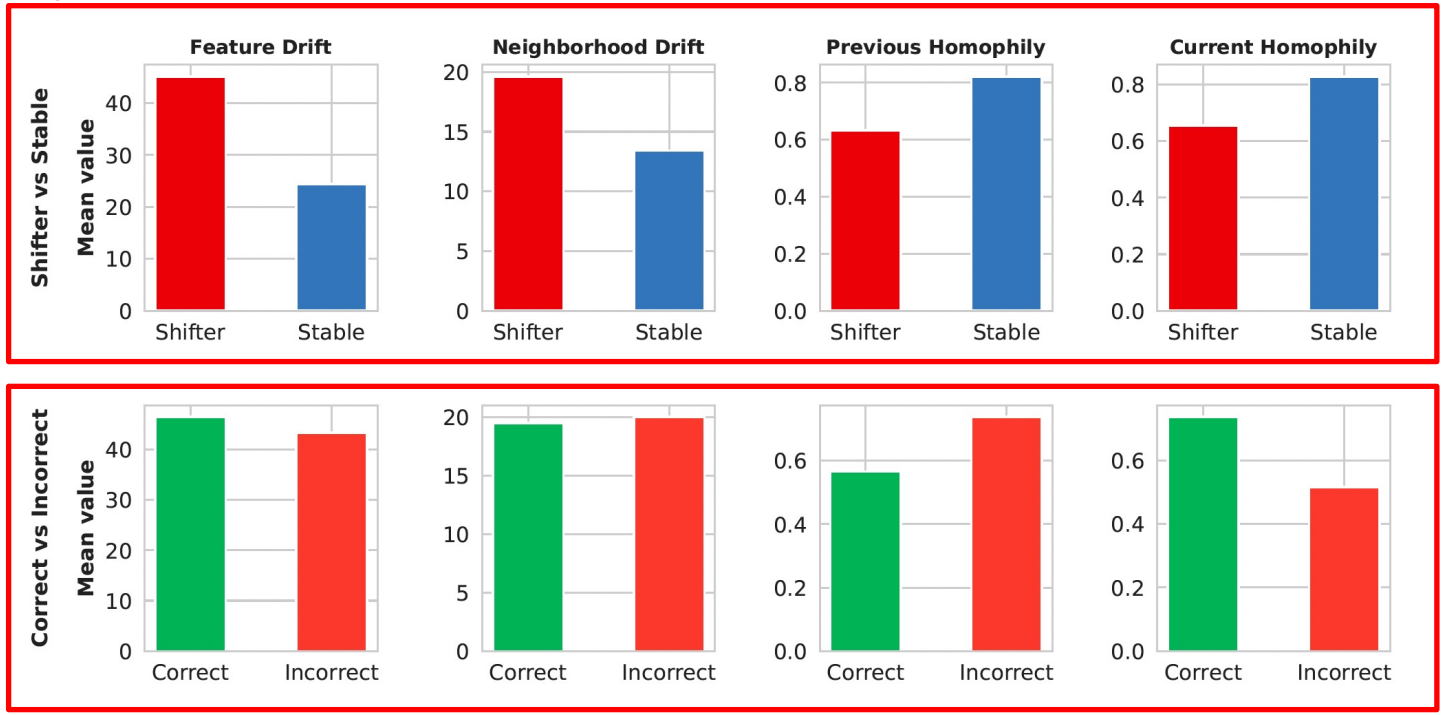


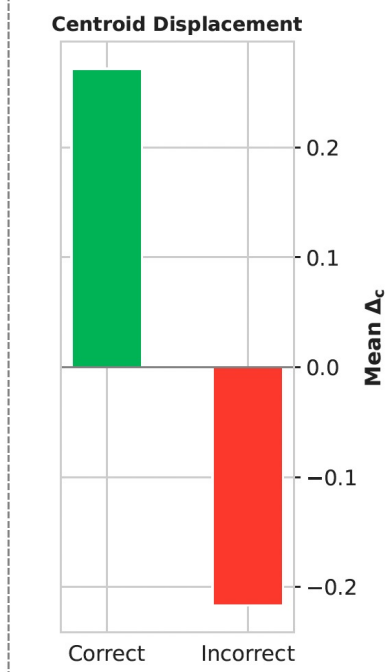
- (Left) 7 dynamic GNNs × 3 datasets: (shifter – stable) accuracy gap up to 50%p
- (Right) **Errors are not random**: on Music (5 classes), up to 76% of shifter errors predict the **old label**
  - 25% under uniform random error

Diagnosis: What Separates Correct from Incorrect?

What separates shifters from stable nodes?

$$\Delta_c(v,t) = \mathrm{sim}(\mathbf{h}_v^{(t)}, \mathbf{c}_{v,\mathrm{new}}^{(t)}) - \mathrm{sim}(\mathbf{h}_v^{(t)}, \mathbf{c}_{v,\mathrm{old}}^{(t)})$$



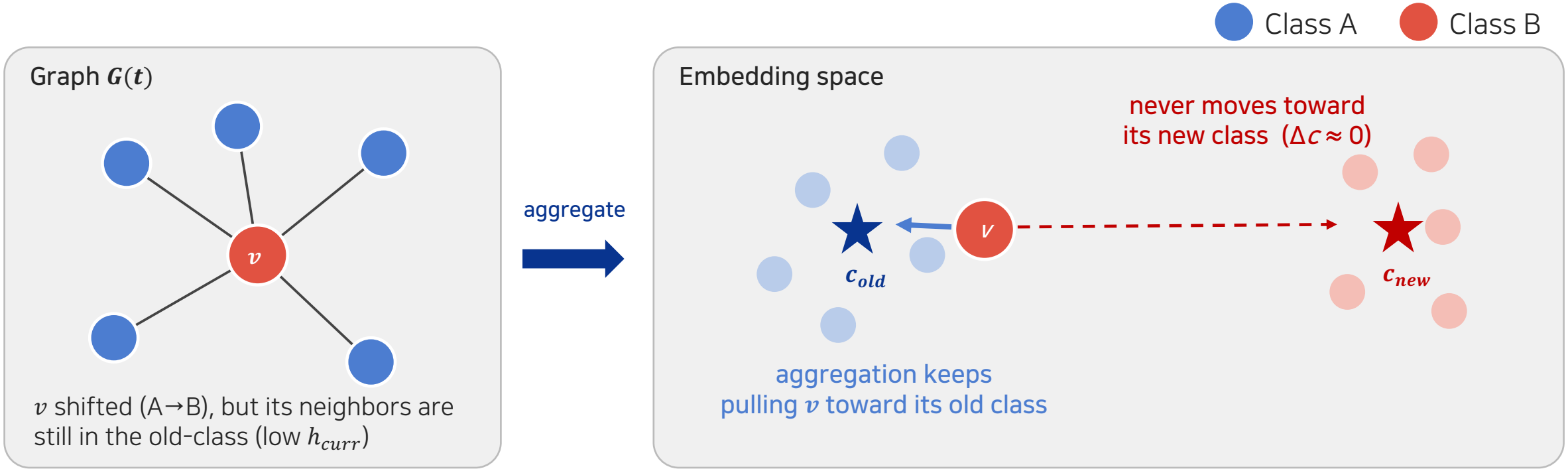


Among shifters only, what separates the ones models get right from the ones they get wrong?

- (Top row) Drift and homophily signals all tell shifters apart from stable nodes — **detecting change is easy**
- (Bottom row) But within shifters, only homophily predicts correctness: wrong shifters have **low current + high previous homophily**
- (Right) Correct shifters move toward the new-class centroid; **incorrect shifters do not move at all**

The Root Cause: Embedding Inertia

Embedding inertia: the representation stays anchored to the old class → **prediction defaults to the old label**



- **Three actionable insights** → the design of CHASE (next)

a Counteract inertia

Pull shifter embeddings toward the new class — strongest where homophily is least supportive

b Detect the shift

Structural signals reliably flag shifters — give the model a dedicated detection head

c Enable the override

Give the classifier an explicit change signal and the capacity to deviate from its past

CHASE: CHange-Aware Framework for Shifting NodEs

- **Model-agnostic wrapper**: keeps any dynamic GNN encoder untouched — replaces only the classifier head
- Three components, one per diagnostic insight: (a) homophily-guided contrastive loss, (b) change detection head, (c) change-aware classifier

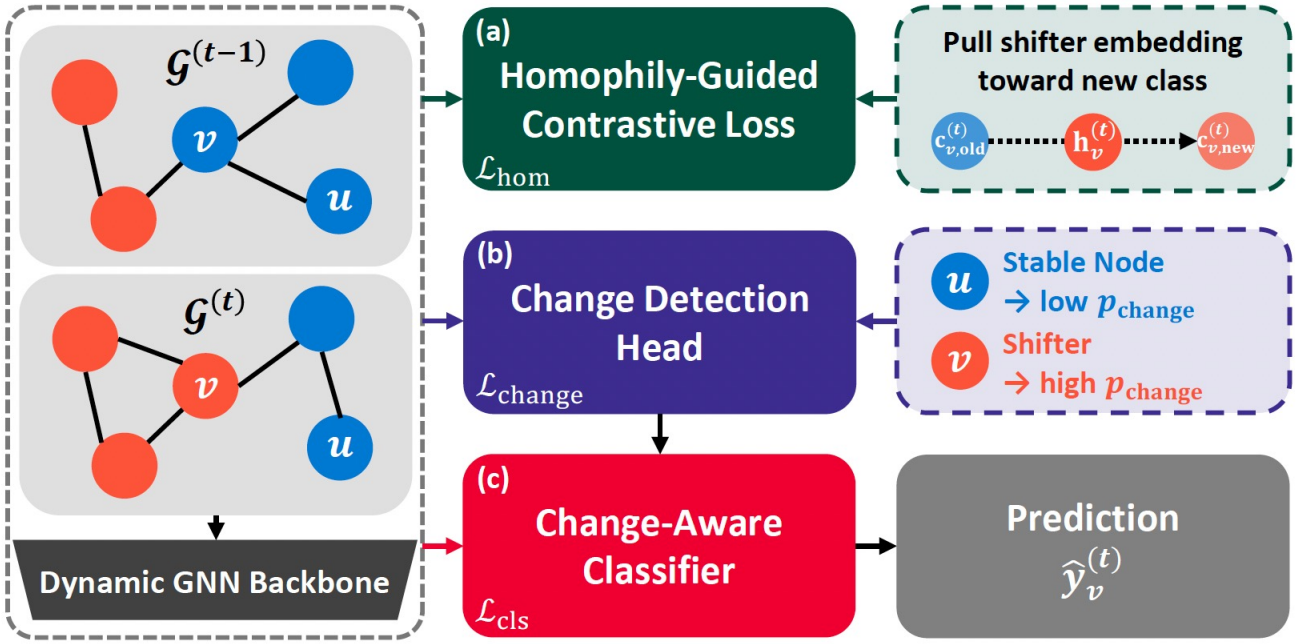


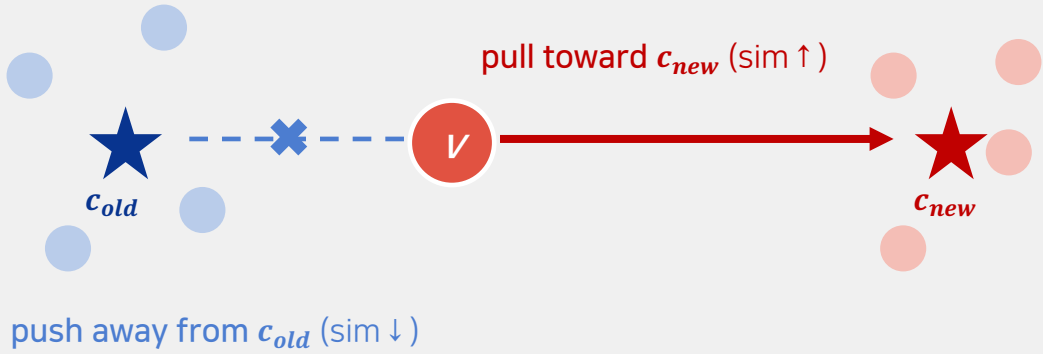
Figure 4: The CHASE architecture. Given snapshots $\mathcal{G}^{(t-1)}, \mathcal{G}^{(t)}$ and a dynamic GNN backbone, the **(a) homophily-guided contrastive loss** pulls shifter embeddings toward their new-class centroid. Then, the **(b) change detection head** estimates the probability p_{change} that a node has shifted. Finally, the **(c) change-aware classifier** receives change signals to guide its prediction.

(a) Homophily-Guided Contrastive Loss

Goal: actively move each shifter's embedding during training — break the inertia that aggregation creates

$$\mathcal{L}_{\text{hom}} = \frac{1}{|\mathcal{S}|} \sum_{(v,t)\in\mathcal{S}} (1 - h_{\text{curr}}(v,t)) \cdot \max\big(0,\ m - (\text{sim}(\mathbf{h}_v^{(t)}, \mathbf{c}_{v,\text{new}}^{(t)}) - \text{sim}(\mathbf{h}_v^{(t)}, \mathbf{c}_{v,\text{old}}^{(t)}))\big)$$

Embedding space (training)



Pull strength ∝ $(1 - h_{curr})$

Reliable anchors

centroids computed from stable-node embeddings — their labels did not change

Adaptive correction

weight $(1 - h_{curr})$: the less the neighborhood helps, the harder the pull

Training-only

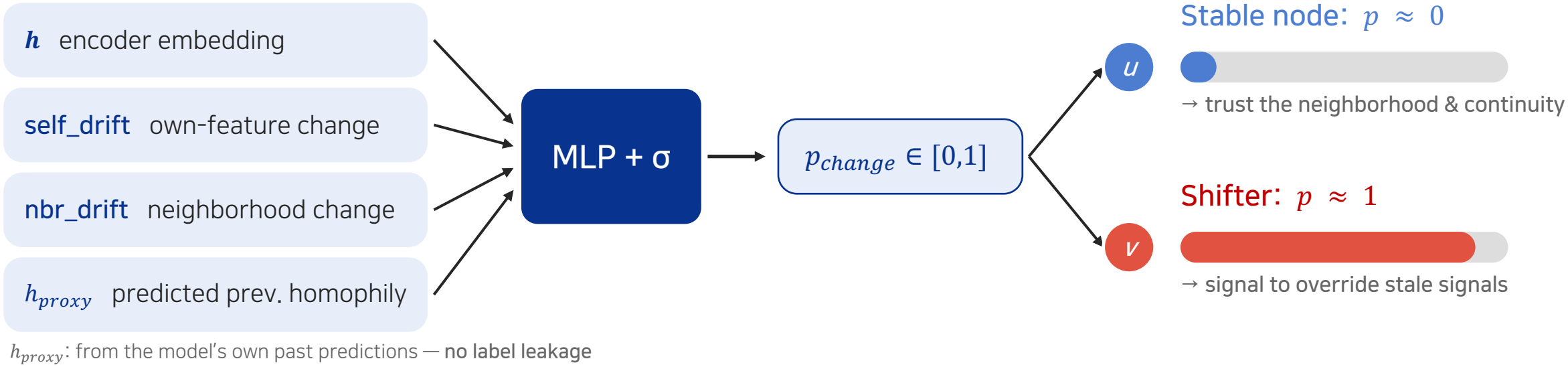
applied only during training — no label information needed at inference

Pull shifters toward where they're going — not where they've been

(b) Change Detection Head

Goal: turn the shift-detection power of structural signals into an explicit probability the model can use

$$p_{\text{change}}(v,t) = \sigma\big(\text{MLP}([\mathbf{h}_v^{(t)};\ \text{self\_drift}_v^{(t)};\ \text{nbr\_drift}_v^{(t)};\ h_{\text{proxy}}(v,t)])\big)$$



h_{proxy}: from the model's own past predictions — **no label leakage**

No label leakage

h_proxy uses the model's own predictions — same recipe at train / val / test

Handles shifter rarity

class-weighted BCE — shifters are only 2.6–18.8% of nodes

Feeds the classifier

a strong detector on its own — its output enters (c) as a soft change signal

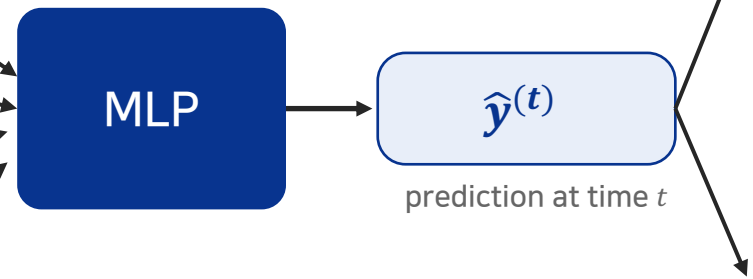
Tell the model that a shift happened

(c) Change-Aware Classifier

Goal: give the classifier the signal for **when** to override prior predictions

$$\hat{\mathbf{y}}_v^{(t)} = \mathrm{MLP}\big([\mathbf{h}_v^{(t)};\ \mathbf{z}_{\mathrm{self},v}^{(t)};\ p_{\mathrm{change}}(v,t);\ \hat{\mathbf{y}}_v^{(t-1)}]\big)$$

- $\boldsymbol{h}$ neighbor-aggregated embedding
- $\boldsymbol{z_{self}}$ aggregation-free own features
- p_{change} soft change signal from (b)
- $\hat{\boldsymbol{y}}^{(t-1)}$ previous prediction



$p_{change} \approx 0 \rightarrow$ continuity

all signals agree:
"rely on $\boldsymbol{h}$ and the previous prediction $\hat{\boldsymbol{y}}^{(t-1)}$"

$p_{change} \approx 1 \rightarrow$ override

$\boldsymbol{h}$ and $\hat{\boldsymbol{y}}^{(t-1)}$ may point to old class:
"rely on $\boldsymbol{z_{self}}$ and deviate"

$\boldsymbol{z_{self}}$: evidence untainted by stale neighbors

Clean self-evidence

z_{self} bypasses neighborhood aggregation — the escape hatch from inertia

Learned end-to-end

$$\mathcal{L} = \mathcal{L}_{\mathrm{cls}} + \lambda_{\mathrm{hom}} \cdot \mathcal{L}_{\mathrm{hom}} + \lambda_{\mathrm{change}} \cdot \mathcal{L}_{\mathrm{change}}$$

Test-time safe

all inputs come from features, structure, and the model's own predictions

Know when to trust the graph — and when to override it

Experimental Setup

- **Backbones (7 dynamic GNNs):** GCN-GRU, EvolveGCN, DySAT, GCN-SE, IDGNN, TADGNN, SiGNN
  - each evaluated with and without CHASE
- **Static baselines:** GCN and EERM (invariant learning for distribution shift), in collapsed and sequential variants
- **Protocol:** temporal train/val/test windows, 5 random seeds, model selection on validation F1
- **Metrics:** accuracy and macro F1, reported separately for shifters and stable nodes

Key question: Does the shifter gap close
without sacrificing stable-node performance?

Main Results: Universal Improvement

Shifter — Stable nodes

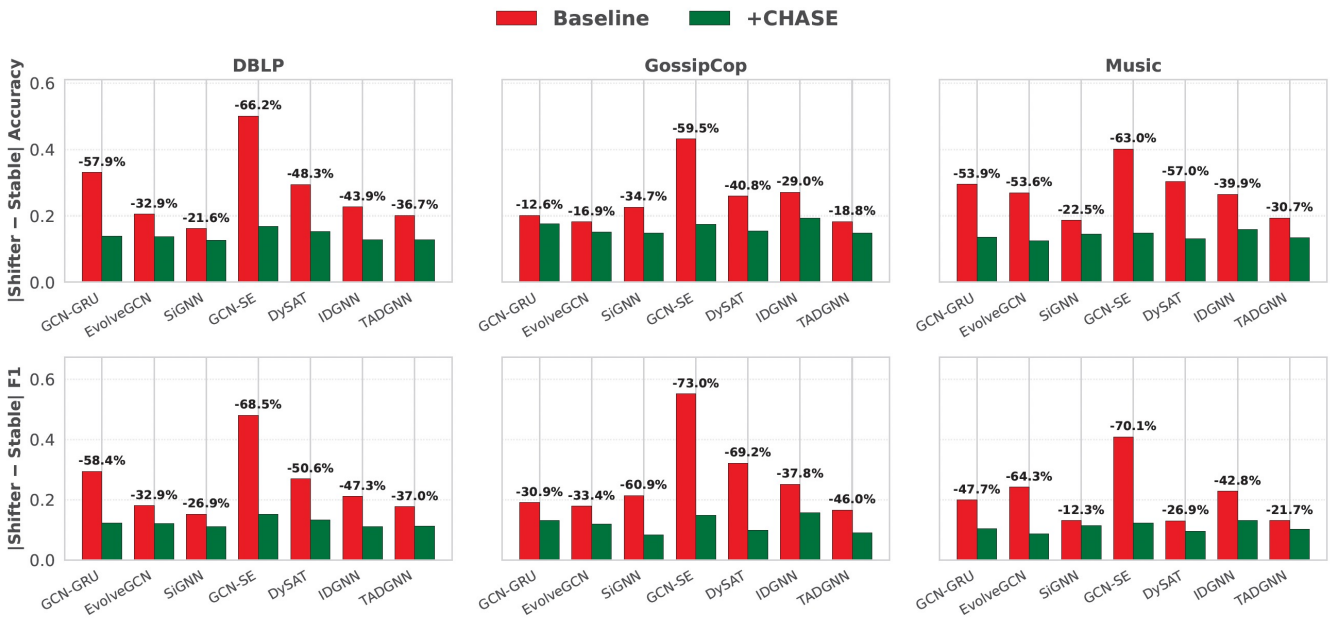
Model	DBLP Sh.Acc	DBLP Sh.F1	DBLP St.Acc	DBLP St.F1
GCN (collapsed)	73.0	76.5	96.0	97.1
GCN (seq)	71.8	75.5	96.1	97.1
+CHASE	**83.2**$^{+15.9\%}$	**85.9**$^{+13.8\%}$	**96.2**$^{+0.1\%}$	**97.2**$^{+0.1\%}$
EERM (collapsed)	73.5	77.4	96.2	97.3
EERM (seq)	73.6	77.3	96.3	97.3
+CHASE	**83.1**$^{+12.9\%}$	**85.6**$^{+10.7\%}$	96.1$^{-0.2\%}$	97.2$^{-0.1\%}$
GCN-GRU	63.9	68.4	97.0	97.8
+CHASE	**82.3**$^{+28.8\%}$	**85.0**$^{+24.3\%}$	96.3$^{-0.8\%}$	97.2$^{-0.6\%}$
EvolveGCN	75.7	79.2	96.2	97.2
+CHASE	**82.4**$^{+8.9\%}$	**85.1**$^{+7.5\%}$	96.2$^{+0.0\%}$	97.2$^{+0.0\%}$
SiGNN	71.3	75.3	87.4	90.6
+CHASE	**83.2**$^{+16.7\%}$	**85.8**$^{+13.9\%}$	**95.9**$^{+9.7\%}$	**97.0**$^{+7.1\%}$
GCN-SE	47.1	49.9	97.2	97.9
+CHASE	**79.5**$^{+68.8\%}$	**82.2**$^{+64.7\%}$	96.4$^{-0.8\%}$	97.3$^{-0.6\%}$
DySAT	61.2	67.1	90.7	94.0
+CHASE	**81.2**$^{+32.7\%}$	**84.1**$^{+25.3\%}$	**96.5**$^{+6.4\%}$	**97.4**$^{+3.6\%}$
IDGNN	73.3	75.9	96.0	97.1
+CHASE	**83.3**$^{+13.6\%}$	**86.0**$^{+13.2\%}$	**96.0**$^{+0.0\%}$	**97.1**$^{+0.0\%}$
TADGNN	76.6	79.9	96.7	97.6
+CHASE	**83.2**$^{+8.7\%}$	**85.9**$^{+7.5\%}$	96.0$^{-0.8\%}$	97.0$^{-0.6\%}$

*DBLP shown; GossipCop & Music in paper

- CHASE improves shifter performance in **all 21 backbone–dataset combinations** — while stable-node performance is preserved or improved
- **EERM ≈ GCN on shifters:** the shifter problem is temporal inertia within the observed graph, **not distribution shift** — invariant learning does not solve it
- Even static sequential GCN benefits immensely → the fix is about **change awareness**, not temporal architecture

The Shifter–Stable Gap Narrows

<Shifter-Stable Gap>



<Ablation Study>

Config	DBLP		GossipCop		Music	
	Sh.Acc	Sh.F1	Sh.Acc	Sh.F1	Sh.Acc	Sh.F1
Baseline	$63.9_{\pm 1.0}$	$68.4_{\pm 1.1}$	$58.2_{\pm 4.7}$	$34.1_{\pm 12.4}$	$50.1_{\pm 7.0}$	$35.2_{\pm 13.6}$
+Contr.	$65.9_{\pm 0.6}$	$69.9_{\pm 0.6}$	$60.4_{\pm 4.8}$	$45.5_{\pm 3.4}$	$53.6_{\pm 7.0}$	$44.6_{\pm 7.4}$
+Hom.wt.	$68.2_{\pm 1.0}$	$71.9_{\pm 1.1}$	$61.9_{\pm 4.5}$	$48.7_{\pm 3.7}$	$58.1_{\pm 4.6}$	$51.7_{\pm 4.5}$
+Cls.	$70.7_{\pm 1.1}$	$73.3_{\pm 1.8}$	$63.4_{\pm 3.6}$	$51.3_{\pm 3.6}$	$62.2_{\pm 4.2}$	$56.7_{\pm 3.6}$
+Detect.	$76.6_{\pm 1.8}$	$80.2_{\pm 2.2}$	$68.4_{\pm 1.4}$	$59.2_{\pm 1.4}$	$76.5_{\pm 1.0}$	$75.3_{\pm 1.0}$
Full	$\mathbf{82.3}_{\pm 0.5}$	$\mathbf{85.0}_{\pm 0.6}$	$\mathbf{70.1}_{\pm 0.9}$	$\mathbf{64.4}_{\pm 1.5}$	$\mathbf{80.1}_{\pm 1.0}$	$\mathbf{79.2}_{\pm 1.0}$

- CHASE shrinks the shifter–stable performance gap by up to 73% across backbones and datasets
- **Ablations:** every component contributes monotonically — detection signal, self-features, and contrastive pull are complementary, not redundant

The Hardest Cases Benefit the Most

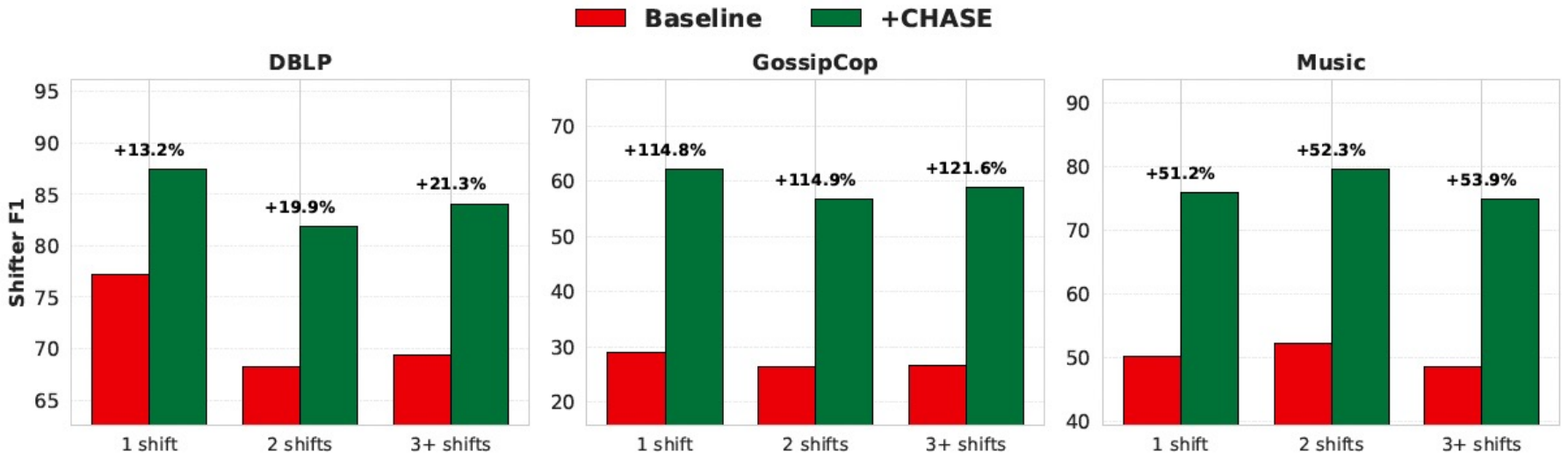


Figure 11: Relative improvement in shifter F1 (+CHASE over baseline), stratified by number of label shifts during test snapshots. Averaged across all seven dynamic backbones. CHASE's benefit increases with shift frequency across all datasets.

- **Setup:** Stratify shifters by how many times they shift during the test window **(1 / 2 / 3+ shifts)**, averaged across all 7 backbones
- **Obs1)** Baseline F1 drops as nodes shift more often — **frequently shifting nodes are the hardest to classify**
- **Obs2)** CHASE's gain grows in the same direction (up to **+121.6%** for 3+ shifts) → **the entities that change the most are helped the most**

Better Than Retraining — at a Fraction of the Cost

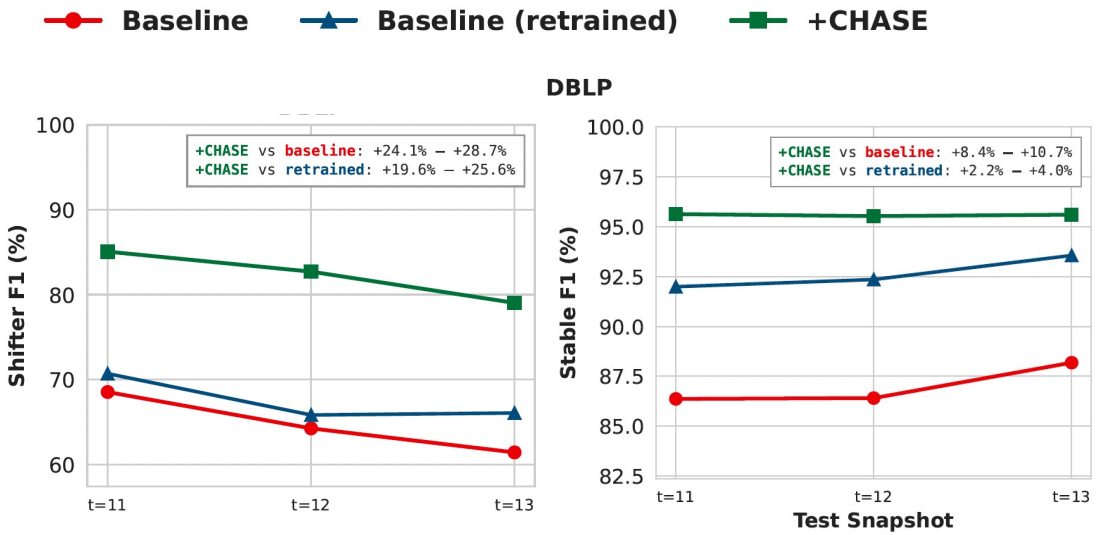


Table 9: CHASE overhead: additional wall-clock time (seconds per epoch) and peak GPU memory (MB) over the baseline.

	DBLP		GossipCop		Music	
Backbone	Δs/ep	ΔMB	Δs/ep	ΔMB	Δs/ep	ΔMB
GCN-GRU	+0.24	+25.6	+0.35	+6.8	+0.24	+2.2
EvolveGCN	+0.40	+42.5	+0.50	+12.7	+0.22	+7.4
SiGNN	+0.25	+137.9	+0.37	+10.5	+0.24	+3.1
GCN-SE	+0.30	+42.9	+0.38	+18.4	+0.21	+3.1
DySAT	+0.24	+184.9	+0.43	+85.2	+0.19	+8.8
IDGNN	+0.14	+10.6	+0.29	+3.4	+0.23	+0.5
TADGNN	+0.33	+185.8	+0.36	+88.5	+0.23	+10.5

- (Left) A strictly stronger baseline: re-train the backbone before every test snapshot on all data so far — including ground-truth labels
- CHASE still wins at every test snapshot, on all datasets — using only its own previous predictions

- (Right) CHASE costs only ~0.24–0.50 s per epoch (< 1 min over full training)

Takeaways for AI for Fraud & Abuse

1. Fraud evolves

Focus on transitions, not just static states.

2. Averages hide risk

Evaluate shifters separately from stable entities.

3. Dynamic ≠ change-aware

Models need to detect change and override stale signals.

4. Early detection matters

The biggest payoff comes from catching entities that just turned bad.

The key question is not only ‘Who is bad?’ but ‘Who just became bad?’

Thank You!

Junghoon

Junmo

Seungyoon

Hyunsung

Yeonjun

Kanghoon

Appendix: Analysis on p_{change}

Table 5: AUROC (%) of p_{change} for shifter detection on test snapshots.

Backbone	DBLP	GossipCop	Music
GCN-GRU	93.1	97.0	79.5
EvolveGCN	87.2	97.6	77.9
SiGNN	83.7	96.9	79.1
GCN-SE	87.3	96.9	78.8
DySAT	90.4	97.0	79.2
IDGNN	85.6	97.4	79.1
TADGNN	84.3	97.1	81.1

p_{change} reliably detects shifters

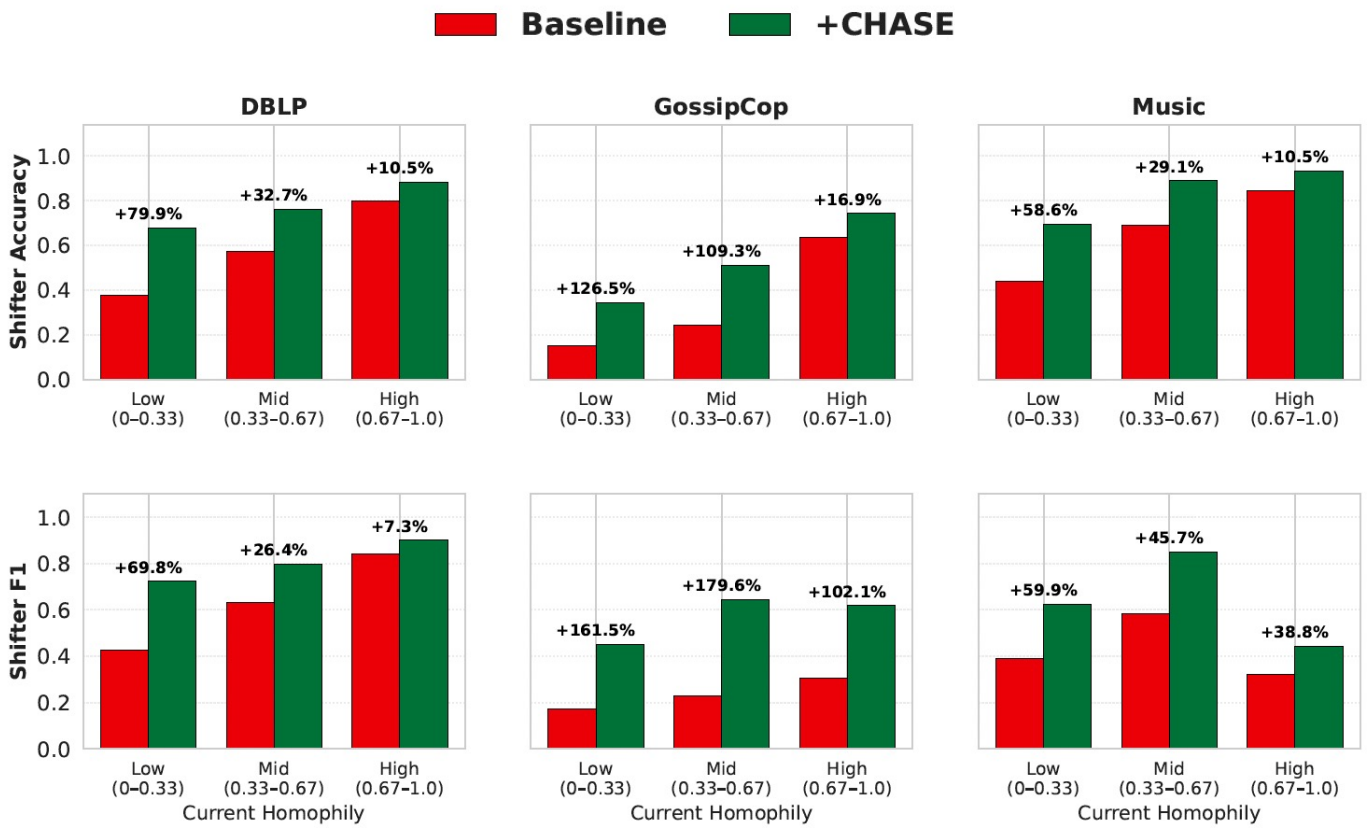
*Rule-based: Flip the label if $p_{change} > 0.5$

Table 6: Shifter F1: CHASE vs. rule-based variant, averaged across backbones.

Method	DBLP	GossipCop	Music
Baseline	70.8	30.1	46.8
Rule-based	74.7	51.4	18.6
+CHASE	**84.9**	**62.2**	**78.4**

- The naive flip rule already beats the baseline — p_{change} works
- But it collapses on Music — knowing ***when*** is not enough without ***what***
- CHASE learns both when to deviate, and how

Appendix: CHASE improvement aligns with homophily



- CHASE's relative improvement on shifter is largest for nodes with the lowest current homophily and decreases monotonically as homophily increases.