TG-Any: Train Once, Predict Links on Unseen Temporal Graphs

Anonymous authors
Paper under double-blind review

Abstract

A new temporal graph should not require a new predictor. We introduce TG-Any, a train-once approach to temporal link prediction through a shared interface of 24 streaming behavioral scores. An 18,584-parameter network learns query-dependent, nonnegative fusion on `tgbl-wiki`; its parameters and source-fitted normalization maps remain fixed on new graphs. A post-hoc pipeline correction and complete matched rerun retain strong transfer relative to source-trained logistic regression and LightGBM: TG-Any reaches 0.1614 mean reciprocal rank on Review and 0.2379 on GoogleLocal, respectively 6.2× logistic regression's MRR and 23.7% above LightGBM. On Coin's official split, its 0.8057 is close to a target-window-trained logistic regression's 0.8056, with an interval containing zero for their difference. Simpler fusion rules qualify these gains: uniform weights reach 0.3133 on Review, and Global is close to TG-Any on GoogleLocal. Global also exceeds query conditioning on the three target validation splits. The shared signal interface supports cross-graph reuse, while greater source accuracy and query conditioning do not guarantee better transfer.

1 Introduction

A new interaction graph should not require learning everything again. An encyclopedia editor choosing a page, a customer reviewing a business, and an account making a transfer involve different entities. Yet useful questions recur: have these entities interacted recently, how often do they interact, and do their histories overlap? Temporal link prediction turns those histories into a ranking of likely next interactions. The challenge we address is to learn a predictor once and carry it to a new graph, without fitting its parameters or feature transforms again.

Simple temporal heuristics provide a promising starting point. EdgeBank, LocalGlobal, and Base3 show that recency, popularity, and neighborhood patterns can compete with learned temporal graph models on established benchmarks (Poursafaei et al., 2022; Cornell et al., 2025; Kondrup, 2025). Their definitions carry across graphs even when node identities, interaction volumes, and score distributions change. Recent contact describes a pair's history; popularity and shared activity describe potential partners, including those with no direct history. These questions establish a common prediction input space. The opportunity is to learn how to interpret and combine their answers once, then reuse that rule on another graph.

We introduce **TG-Any**, a temporal link predictor that learns over this shared temporal-signal interface. A bank of 24 streaming heuristics describes each candidate interaction, source-fitted quantile maps put the scores on a common scale, and an 18,584-parameter network chooses nonnegative weights for each query. The design builds on GraphAny's analytical-channel fusion (Zhao et al., 2024), bringing together causal interaction histories, candidate scoring, and candidate-distribution conditioning for temporal prediction. The new graph supplies its own history, while the parameters and normalization maps remain fixed. What transfers is a rule for scoring interactions; no learned node identity needs to be shared between graphs.

The corrected comparisons show useful reuse alongside substantial limits. On official splits, TG-Any reaches 6.2× the MRR of source-trained logistic regression on Review and improves over source-trained LightGBM by 23.7% on GoogleLocal, using matched candidate features. On Coin, its 0.8057 MRR is close to the 0.8056 of logistic regression fitted to 250,017 target-training events; the paired

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

interval includes zero and only two of five TG-Any seeds exceed that control. Global weights exceed query conditioning on the three target validation splits, while uniform weights lead on Review validation and its official split (0.3133 versus 0.1614). Conditioning and Global have an uncertain difference on GoogleLocal.

These estimates come from a versioned, post-hoc correction of feature construction, source normalization, and numerical execution, followed by refitting and full matched scoring. Previously inspected official splits are not new untouched confirmation. We retain the original experiments and their decisions separately, and report every fixed seed and valid negative comparison (Appendix C). The findings support cross-graph reuse without establishing that a more specialized source predictor transfers better.

Our contributions are:

1. **A train-once temporal prediction method.** TG-Any combines causal temporal signals, source-fitted normalization, and query-dependent nonnegative fusion into a complete predictor reusable across graphs without target fitting.
2. **Cross-graph reuse from a single source.** The same frozen predictor improves over the three original source-trained same-feature controls on review and GoogleLocal, and reaches comparable accuracy to a target-window-trained control on coin.
3. **A controlled characterization of transferable fusion.** Uniform, objective-matched linear, and global-mixture comparisons show that useful transfer depends on how shared signals are combined, beyond source accuracy alone; query conditioning improves the source but does not beat global weights on the three target validation splits. Simple fusion rules substantially strengthen the transfer baselines.

2 Related Work

From strong heuristics to learned mixtures. TGB (Huang et al., 2023) and TGB-Seq (Yi et al., 2025) make temporal link prediction comparable across interaction graphs, including settings with little edge repetition. EdgeBank (Poursafaei et al., 2022) demonstrates the strength of memorization, while LocalGlobal combines local and global recency and popularity (Cornell et al., 2025). Base3 (Kondrup, 2025) combines EdgeBank memorization, PopTrack destination popularity (Daniluk & Dąbrowski, 2023), and temporal co-occurrence memory. Its candidate-dependent confidence rules select among hand-set mixtures. TG-Any instead learns both the mixture and the rule for choosing its weights, then reuses that rule on unseen graphs. Interpretability studies also connect temporal graph models to these signal families (Hayes et al., 2025).

Learning across graphs. GraphAny (Zhao et al., 2024) learns attention over analytical linear graph neural network (LinearGNN) predictions for inductive node classification, providing a precedent for learned signal fusion. TG-Any builds on this idea with causal interaction scores and candidate-distribution summaries for temporal link prediction. Other transfer methods align features across pretraining graphs (Hou et al., 2024) or map structural roles into TGN memory embeddings (Chatterjee et al., 2025). Related temporal models explore different transfer settings: MiNT pretrains across token networks for zero-shot graph-property prediction, including transfer to social networks (Shamsi et al., 2025); TGPM transfers link predictors without target adaptation (Ma et al., 2026); and POSTRA studies zero-shot reasoning on relation-typed temporal knowledge graphs (Pan et al., 2025a). Closer work directly studies cross-graph dynamic link prediction. CrossLink learns across multiple source graphs and conditions a decoder-only Transformer on a target graph's historical evolution sequence, without further parameter training on that graph (Huang et al., 2025). CrossDyG directly studies single-source cross-graph dynamic link prediction through deconfounded representation learning and target-specific structural intervention (Pan et al., 2025b). TG-Any addresses this shared goal through an explicit temporal-signal interface: candidate interaction histories become behavioral scores, source quantile maps set their scale, and a query-dependent network learns their nonnegative fusion. This concentrates learning on how to combine reusable signals, using timed interactions without node or edge attributes. Our experiments compare fusion choices under the same interface and benchmark candidate sets.

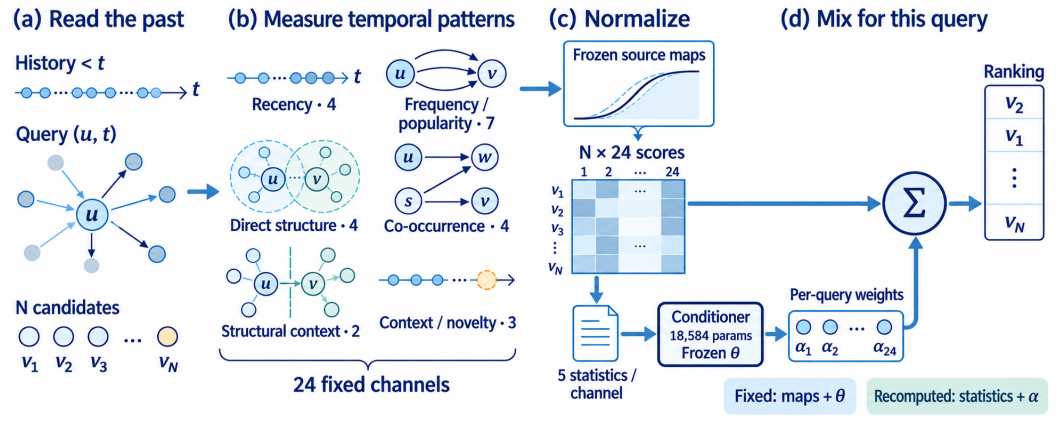


Figure 1: **Learn how to combine patterns, then reuse the predictor.** Past interactions supply 24 heuristic scores for each candidate. Source-fitted maps put them on a common scale. A small conditioner reads five summaries per channel (mean, standard deviation, maximum, effective candidate fraction, and distinct-score gap) and chooses query-specific weights for the final weighted sum. Signal definitions, maps, and network parameters stay fixed on new graphs; history, candidate scores, summaries, and mixture weights are recomputed for each query. Networks, score matrices, and rankings illustrate the computation.

Expressive temporal models and conditioning. TGN, DyGFormer, and JODIE learn temporal representations from event streams (Rossi et al., 2020; Yu et al., 2023; Kumar et al., 2019). TNCN and SDG further improve prediction through neural structural modeling and sequence diffusion (Zhang et al., 2024; Nguyen Minh Duc & Viet Cuong Ta, 2026). Candidate-conditioned scoring also appears in next-location recommendation (Yu et al., 2026). TG-Any conditions a single mixture on the score distribution of the whole query, so all candidates share the resulting channel weights. Our experiments focus on the value of this learned predictor under a matched feature bank; published in-domain neural results provide broader performance context.

3 TG-Any: A Predictor Built from Reusable Signals

3.1 Predicting the next interaction

A temporal graph is a stream of interactions (u, v, t): source node u interacts with destination v at time t. Given a source and a time, we rank the true destination against the dataset's candidate alternatives. A query q has an ordered candidate list $\mathcal{C}$ with $N = Q(q) + 1$ positions: one positive and $Q(q)$ supplied negatives. Duplicate negative destinations retain their positions. The predictor can use only earlier events. We therefore score every query at a timestamp before adding that timestamp's events to history.

TG-Any follows three steps (Figure 1): measure familiar patterns in the history, normalize their scores, and learn how much each pattern should matter for this query. The same operations apply to the source and every target graph.

3.2 A shared temporal-signal interface

The bank contains 24 heuristics in six families: recency, frequency and popularity, direct neighborhood similarity, co-occurrence similarity, structural context, and context or novelty. Together they describe when nodes were active, whether a pair has interacted before, and how its interaction histories relate. These definitions use events and local history, so they apply without carrying node embeddings from one graph to another. For example, pair recency asks the same question about two accounts as it does about an editor and a page. Candidate-varying channels distinguish destinations directly. Source-only channels, such as source recency and activity, instead provide query context that the conditioner can use to allocate weight across scoring channels.

Structural scores need particular care on bipartite graphs. An editor's neighbors are pages, while a page's neighbors are editors; their direct neighbor sets cannot overlap. Our co-occurrence channels take an extra hop through a candidate's recent sources to reach other destinations. They can then compare those destinations with the query source's own history. This supplies structural information even when direct overlap is always zero. Both neighborhood definitions remain fixed across graphs. Appendix A lists every channel and the bounded history rules.

Making scores comparable. A lifetime count and a time-gap score have very different scales. In the corrected pipeline, channel 24 uses the Wiki training statistic $g_{\text{src}} = 300$ in Wiki's provided timestamp units, transferred unchanged to every target. TG-Any maps each channel into $[0, 1]$ using empirical quantiles fitted on source training data. The corrected maps use one common two-million-row sample selected by stable candidate-identity SHA256 across all 24 channels. High and low values can then be combined without letting a channel dominate simply because of its units. The same maps are reused on every target. With this frozen wiki fit, the four direct structural channels are constant; their behavior and the exact normalization rule are documented in Appendix A, with correction provenance in Appendix C. Consequently, target direct overlap cannot contribute to rankings through those four channels under this source fit.

3.3 Choosing a mixture for each query

A useful mixture should respond to the evidence in front of it. A channel that gives one candidate a clear lead presents different information from a channel that ties every candidate. For each channel, TG-Any summarizes the candidate scores by their mean, spread, maximum, concentration, and the gap between the two highest distinct values. A two-layer network reads these summaries and outputs 24 nonnegative weights that sum to one. All candidates in the query share these weights:

$$s(v_i \mid u, t, \mathcal{C}) = \sum_{k=1}^{24} \alpha_k(u, t, \mathcal{C})\, \widetilde{x}_{ik}, \tag{1}$$

where $\widetilde{x}_{ik}$ is candidate i's normalized score on channel k, and α_k is the weight assigned to that channel for candidate set $\mathcal{C}$. The final ranking orders candidates by s. Nonnegative shared weights preserve within-query channel dominance: a candidate that scores at least as high on every channel cannot be ranked below the candidate it dominates.

The conditioner uses a fixed number of summaries even when candidate sets differ in size, and receives no explicit candidate count. Appendix A gives their exact definitions and invariance properties. Our ablations compare this design with explicit count input and with one learned mixture shared by all queries.

Learning once. The network has hidden width 128 and 18,584 parameters. We train it on source queries with listwise InfoNCE (van den Oord et al., 2018), which rewards ranking the observed destination above its negative candidates, using Adam (Kingma & Ba, 2014). Once training and source checkpoint selection are complete, the network and quantile maps are fixed. On a target graph, the heuristic history advances as events arrive and the predictor computes fresh candidate scores, summaries, and mixture weights for each query. Signal definitions, normalization maps, and network parameters stay fixed. This is query-dependent inference without target fitting, the strict zero-shot setting used for all TG-Any transfer evaluations. Exact feature and training settings appear in Appendices A, C, and I.6.

4 Experimental Setup

Our experiments ask whether a mixture learned on one temporal graph can predict links on another, and which design choices affect that transfer. The source is `tgbl-wiki-v2`. The targets span reviews, local business interactions, token transfers, and flights from TGB and TGB-Seq (Huang et al., 2023; Yi et al., 2025). Dataset sizes and splits are listed in Table 8 and Appendix F.

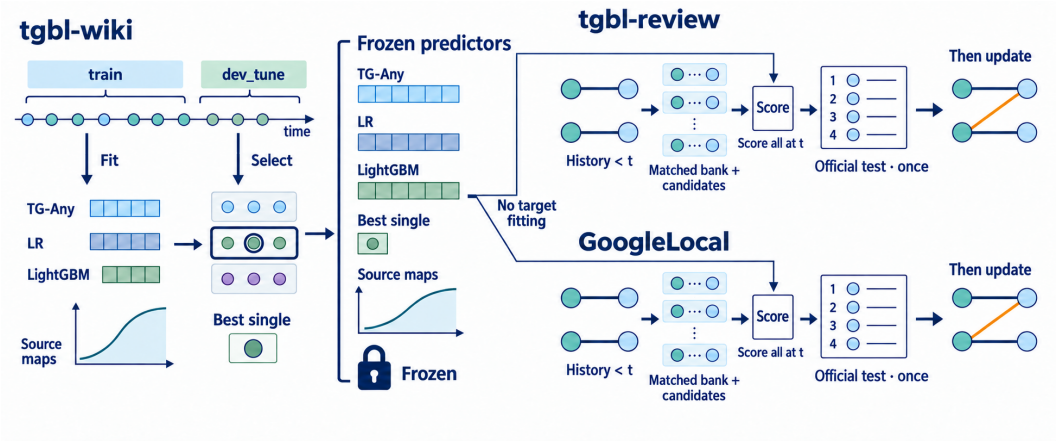


Figure 2: **Original source-only transfer setup.** Wiki training fits the predictors and quantile maps; wiki `dev_tune` selects TG-Any checkpoints and the best single channel. The frozen predictors then score review and GoogleLocal using each target's past history and matched candidates. History is updated after all queries at the timestamp are scored. The later coin and flight evaluations compare the same frozen TG-Any with LR and LightGBM fitted on specified target-training windows, and raw BestSingle selected on the first 50,000 validation queries. All networks and ranks are schematic.

4.1 Train on the source, evaluate on new graphs

We train on wiki's official training split. Its validation split is divided chronologically into `dev_tune` (11,810 queries) for hyperparameter and checkpoint selection, and `dev_gate` (11,811 queries) for a held-out source comparison. Review validation provides a declared development transfer target. In the original campaign, official tests remained unseen during development and were each scored once with the frozen primary predictor. Figure 2 shows that original transfer setup. The corrected analyses are post-hoc rescores after target outcomes were inspected. They preserve the original source-gate and test-entry decisions: neither corrected gate scores nor corrected official-set scores are new untouched confirmation or a second execution of the original one-time test protocol (Appendix C).

The original campaign evaluates wiki, review, and GoogleLocal official tests. We also report coin and flight from a separately preregistered expansion. Its validation-first rule proceeds to a target's official test only when TG-Any exceeds its strongest control. Coin met that condition in the original campaign; Flight did not. This rule conditions official-test coverage on validation performance; the test results do not establish success on all evaluated targets. The official tests use benchmark-provided candidates and MRR, the mean reciprocal rank of the true destination. Sampling and the original GoogleLocal loader repair are documented in Appendix B.

4.2 Comparisons and reporting

Logistic regression and LightGBM (Ke et al., 2017) use the same 24-channel bank and source-fitted normalization as TG-Any. BestSingle instead selects and ranks one raw channel, cast to float64. In the original tests, these controls are trained or selected on wiki and frozen on the targets. This comparison asks whether the learned predictor improves transfer when the channel definitions and training graph are held fixed. In the expansion, LR and LightGBM use fixed hyperparameters and fit the specified target-training windows without validation model selection. Raw BestSingle selects its channel on the first 50,000 target validation queries; it does not fit an estimator. TG-Any still uses only its source fit.

The source comparison additionally includes LocalGlobal and two Base3 reimplementations, giving six controls in total. The preregistered comparison rule appears in Appendix B. That six-control K2 gate remains a historical decision and is not retroactively recomputed from the corrected post-hoc comparisons. We report published in-domain neural results separately to place the transfer results in context.

Table 1: **Official test MRR with fixed source-only models.** Wiki is the source; review and GoogleLocal are unseen targets. TG-Any and Global report five-seed mean ± std; other rows are deterministic. All rows share the bank and each target's candidates. Learned models and Uniform use source-fitted maps; Best single channel uses raw values. Query counts are 728,919 on Review and 43,546 on GoogleLocal. Source official results are reported in Table 10. All values are corrected post-hoc estimates; original results and the query-conditioned primary identity are retained in the audit record. Bold marks the highest mean, not a significance test.

Method	Review (target)	GoogleLocal (target)
Best single channel	0.0197	0.0196
Logistic regression	0.0262	0.1491
LightGBM	0.0228	0.1923
Uniform (post-hoc)	**0.3133**	0.2134
Global mixture (post-hoc)	0.1931 ± 0.0088	0.2369 ± 0.0042
TG-Any (conditioned)	0.1614 ± 0.0010	**0.2379 ± 0.0036**

Supplementary validation compares a uniform average with an unconstrained 24-weight linear ranker trained using the same InfoNCE objective and source-only checkpoint selection as TG-Any. The corrected comparison also evaluates the refitted Global and primary checkpoints under the same frozen corrective recipe. These analyses were added after the original campaign; all configurations were fixed before their target scoring. Details appear in Appendix E.5.

Corrected same-feature comparisons use the pinned Python 3.10.20, NumPy 2.0.2, and Torch 2.5.1 CPU runtime with one numerical thread and batches of 64 complete queries without splitting the candidate axis. Reciprocal ranks use float64 and the exact average-tie rule without a tolerance (Appendix C).

Primary results report mean and standard deviation over five training seeds. The original same-feature controls are deterministic. Conditioner ablations use five seeds; normalization ablations use three and are exploratory. Corrected paired analyses align exact query identities and timestamps, average the prescribed seeds within each query, preserve complete timestamp groups, and use query-count-weighted bootstrap blocks at three prescribed lengths. The resulting confidence intervals are pointwise and conditional on the fitted seeds. Historical records include Holm-adjusted bootstrap sign-tail estimates (Holm, 1979); these are descriptive resampling quantities, not calibrated null-test p-values. Our conclusions use effect sizes, intervals, seed variation, and the stated mean-based evaluation criteria. Full statistical and reproducibility details appear in Appendices B.7, C, and I.

5 Results

TG-Any reuses one wiki-trained predictor across three officially evaluated target graphs. Unless explicitly labelled historical, results below use the corrected features, refitted predictors and controls, and full fixed evaluation cohorts. These are post-hoc comparisons on previously inspected splits. Source and fusion comparisons examine what learning contributes to this reusable interface; original decisions remain separate.

5.1 Transfer benefits and simple fusion baselines

TG-Any outperforms the three original source-trained same-feature controls on both review and GoogleLocal official tests (Table 1). On review, its MRR is 0.1614, compared with 0.0262 for the strongest original control, logistic regression. TG-Any thus reaches 6.2× the control's MRR using the same source graph, normalization, and candidate features. This is an advantage over these fitted controls, not over every way of combining the shared signals.

The strongest original control changes on GoogleLocal, but TG-Any's advantage persists. It reaches 0.2379 MRR versus 0.1923 for LightGBM, a 23.7% relative improvement. Paired temporal block-bootstrap intervals for the gains over the strongest original control exclude zero on both targets (Ta-

Table 2: **Coin official test: source-trained TG-Any against target-window-trained logistic regression.** TG-Any reports five-seed mean ± std. The control fits 250,017 training events (1.5659% of Coin training), with full earlier history. Corrected validation comparisons appear in Table 18; the historical test-entry decisions are documented separately in Appendix F.1.

Predictor	Training graph / fitted window	Coin test MRR
TG-Any	Wiki only	**0.8057 ± 0.0014**
Logistic regression	Coin / 250,017 events	0.8056

ble 12). These tests use the same frozen TG-Any checkpoints. Review has few strictly prior positive-pair repeats; the query-level analysis below tests whether its gain extends beyond those repeats.

The simple fusion baselines qualify this advantage. On Review, Uniform reaches 0.3133 and Global 0.1931, both above TG-Any's 0.1614. On GoogleLocal, TG-Any exceeds Uniform, while its 0.0010 mean advantage over Global has a paired 95% interval of $[-0.00064, 0.00275]$. These results support reusable signals but do not establish query conditioning as the best transfer rule (Table 20).

A direct query-level analysis makes this concrete on review validation. Among pairs that had never interacted before the query timestamp, TG-Any reaches 0.1360 MRR against logistic regression's 0.0363 (Appendix E.9). The improvement therefore extends to predicting new pairings, beyond recalling past edges.

Published in-domain results provide external context. TGN reaches ∼0.349 on Review and ∼0.541 on GoogleLocal (Huang et al., 2023; Yi et al., 2025). The current official Review-v2 leaderboard also lists that Review value. These target-trained results are not a controlled gap estimate: the published evidence does not bind the same candidate ordering, negative-file bytes, evaluator, numerical runtime, and training protocol as this comparison.

5.2 Competitive prediction without target optimization

Coin tests whether source learning remains useful when the controls can learn from the target graph. On its official test, TG-Any achieves 0.8057 MRR versus 0.8056 for target-window-trained logistic regression (Table 2). Two of the five TG-Any seeds exceed that control. The mean difference is +0.000120, with a paired 95% interval of $[-0.000411, 0.000637]$. Thus the observed means are close; the interval does not establish either superiority or statistical equivalence. Coin is non-bipartite and recurrence-heavy, extending the transfer result to a different interaction pattern from review and GoogleLocal.

Flight provides a more demanding transfer case. On validation, TG-Any scores 0.8895, below both target-window-trained LR (0.8941) and LightGBM (0.8927), while exceeding BestSingle (0.8773). The primary-minus-LightGBM difference is -0.003175, with base 95% interval $[-0.003995, -0.002344]$. All three prescribed block lengths retain the loss to both fitted controls. The historical Flight test-entry gate failed; its official split remains unevaluated. These corrective comparisons do not rerun that decision (Appendix F).

5.3 Learning to fuse reusable signals

Both Global and query-conditioned TG-Any outperform uniform weights on Coin and Flight validation (Table 3). On Coin, conditioning raises MRR from 0.7230 to 0.7851, while Global reaches 0.7906. On Review, Uniform leads instead. Learned signal fusion can therefore help transfer, but the benefit depends on the target. The corrected Wiki comparison favors TG-Any over Global and canonical LocalGlobal; it is distinct from the historical six-control gate (Appendix E.1).

Additional source specialization does not uniformly improve transfer. On review validation, uniform weights reach 0.2678 MRR, above the global mixture's 0.1619 and the primary's 0.1366. Source accuracy follows the reverse order: uniform scores 0.3522, global weights reach 0.8383, and conditioning reaches 0.8421. Global weights exceed the conditioned primary on all three target validation splits. The review difference is +0.0253 MRR (95% CI [+0.0246, +0.0260]). Source accuracy alone therefore does not identify the best transfer rule.

Table 3: Corrected fusion comparison on Wiki source gate and three target validation splits. Learned rows report five-seed mean ± sample standard deviation; Uniform has no fitted weights. All methods share corrected features, frozen source quantile maps, and matched candidate positions. Bold marks the highest mean, not a significance decision. This is a post-hoc comparison.

Predictor	Wiki	Review	Coin	Flight
Uniform	0.3522	**0.2678**	0.7230	0.7985
InfoNCE linear	0.8257 ± 0.0019	0.0526 ± 0.0316	0.7784 ± 0.0066	0.8776 ± 0.0103
Global mixture	0.8383 ± 0.0002	0.1619 ± 0.0071	$\mathbf{0.7906 \pm 0.0003}$	$\mathbf{0.8926 \pm 0.0005}$
TG-Any	$\mathbf{0.8421 \pm 0.0003}$	0.1366 ± 0.0005	0.7851 ± 0.0008	0.8895 ± 0.0007

The objective-matched linear control sharpens this result. Its 24 unconstrained weights achieve 0.8257 MRR on wiki, but only 0.0526 on review. It also trails the nonnegative learned mixtures on coin and flight. Using InfoNCE with the shared features is thus not enough to reproduce their transfer. The choice of parameterization matters alongside the training objective (Appendix E.5). TG-Any provides a shared prediction interface in which these fusion choices can be studied without changing the signals or introducing target fitting.

Other design choices. Adding explicit candidate count has a small effect relative to the primary on Wiki and Review (Table 13); on Review the primary's mean is 0.000526 higher, with base interval $[0.000470, 0.000585]$. The matched three-seed normalization study orders source means as quantile, raw, z-score, then log-only. Transfer changes that order (Table 26): quantile leads the four normalization recipes on Coin (0.7849 versus 0.5516 for log-only), while log-only leads on Review (0.1522 versus 0.1366 for quantile). Every map uses its corresponding source-trained checkpoints; none is refitted on a target. The corrected raw-score source runs reach 0.8362, 0.8331, and 0.8331. The earlier low-performing raw seed-0 run remains historical evidence, not a collapse of the corrected model. These comparisons combine representation and optimization effects and do not isolate one repair. Full recurrence, within-query variance, and summary diagnostics appear in Appendix G.6.

5.4 Candidate-set size changes the comparison

TG-Any maintains strong source validation performance as the negative candidate count increases: 0.8765 MRR at $Q = 20$, 0.8576 at $Q = 100$, and 0.8421 at full width (Table 17). LightGBM leads at the two smaller widths, with 0.8931 and 0.8611, but falls to 0.5446 at full width. The relative ordering therefore depends on how many alternatives a predictor must rank. This experiment supports evaluating transfer models with the benchmark's full candidate sets, alongside any reduced-width analysis.

6 Discussion

Shared signals support reuse, with a restricted claim. TG-Any gives interaction graphs a common prediction interface: each graph supplies its causal history, while source-fitted maps and fusion parameters remain fixed. The corrected comparisons show gains over the three original source-trained same-feature controls on Review and GoogleLocal, and similar accuracy to a limited-window target-trained control on Coin. Review's gain over logistic regression occurs mainly among previously unseen positive pairs. These results support useful transfer of the signal interface; they do not establish superiority to all simple rules or to fully target-trained temporal models.

Learning and conditioning have different effects. Learned nonnegative mixtures improve on uniform weights on Coin and Flight validation, but uniform weights substantially exceed both learned mixtures on Review. Conditioning improves source-gate accuracy over Global, yet trails it on all three target validation splits and has no clear advantage on GoogleLocal's official split. Normalization rankings also change across targets. These findings motivate objectives evaluated for cross-graph utility, while the current single-source study cannot identify a generally best fusion rule, transform, or training graph.

What the correction establishes. The corrected pipeline removes documented implementation discrepancies and aligns fits, candidates, runtime, and paired uncertainty estimates. The legacy-to-corrected comparison retains five seeds and the same query cohorts; it does not isolate the effect of one repair. Small changes in primary MRR do not imply that all implementation choices are harmless: the normalization ablation changes materially, and controls must be refitted to maintain a valid comparison. Conversely, a valid loss to Global or Uniform is not evidence of another bug. Channel 24 is constant across candidates within each query but can still affect the conditioner; source-constant direct-overlap channels collapse under the frozen source maps. These limits are measured rather than tuned away.

Scope and future directions. The study uses one training graph and four transfer targets. Coin and Flight's main fitted controls use only 1.5659% and 0.6403% of target training events. Coin's larger-window LR sensitivity has an uncertain difference from TG-Any at the base block length, and its post-selection suffix comparison is sensitive to block length. BestSingle selection and full validation overlap; the reported suffixes retain the original selected channel and are post-hoc sensitivities. Flight's historical failed test-entry decision remains in force. All corrected official-set results follow inspection of earlier outcomes, and pointwise intervals condition on fitted seeds rather than measuring variation across new graphs. On Review validation, the fixed three-seed adapted CrossDyG run reaches 0.045149 MRR versus TG-Any's 0.136552 on the same queries; this architecture comparison remains specific to the disclosed source-only adaptation (Appendix G.3).

The 18,584 parameters describe the base conditioner, not end-to-end cost, which also includes causal state and feature construction. Multi-source objectives, separately defined history-based calibration (Appendix A.2), and broader candidate distributions remain future work. Those experiments require new, prospectively separated evidence rather than reinterpreting this corrective rerun as untouched confirmation.

7 Conclusion

TG-Any reuses temporal prediction through a shared behavioral-signal interface and a source-trained fusion rule. A complete, versioned correction retains gains over the original source-trained same-feature controls on Review and GoogleLocal and comparable mean accuracy to a target-window-trained control on Coin. The same evidence limits the architectural claim: Uniform leads on Review, Global is close on GoogleLocal and exceeds conditioning on the target validation splits, and normalization rankings vary across graphs. These preserved negative comparisons make the interface a concrete setting for studying which prediction rules transfer, without equating source accuracy with cross-graph generalization.

Reproducibility Statement. The appendices document features, fitting, checkpoint selection, evaluation, numerical execution and statistics. Versioned correction records separate current and historical results and preserve failures and unfavorable comparisons. The supplementary code archive includes the corrected implementation, 29 learned checkpoints, seven serialized control models, three BestSingle descriptors, and Uniform. Its manifest links current evidence and external dependencies. Independent extraction checks passed; score-only numerical checks are inherited through byte-identical portable artifacts, without a new numerical rerun. The entry point scores supplied corrected features; full graph-feature and per-query evidence reconstruction requires the listed external inputs.

Ethics Statement. This study uses public TGB and TGB-Seq datasets and involves no new human-subject data collection. Dataset sources and use conditions are documented by the benchmark providers (Huang et al., 2023; Yi et al., 2025). The Base3 comparison reports the outcomes of two reimplementations and the remaining gap from its published result.

AI Use Statement. LLMs assisted with implementation, protocol drafting, artifact review, and writing. GPT Image 2 assisted with Figures 1 and 2; their schematics were checked against the method and protocol. The authors reviewed all AI-assisted work and take responsibility for the content. Numerical results come from experimental code and saved outputs.

REFERENCES

Ayan Chatterjee, Barbara Ikica, Babak Ravandi, and John Palowitch. Transfer learning for temporal link prediction. In *2025 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–8. IEEE, June 2025. doi: 10.1109/ijcnn64981.2025.11228624. URL http://dx.doi.org/10.1109/IJCNN64981.2025.11228624.

Filip Cornell, Oleg Smirnov, Gabriela Zarzar Gandler, and Lele Cao. On the power of heuristics in temporal graphs, 2025. URL https://arxiv.org/abs/2502.04910. arXiv version; published at the ICBINB Workshop at ICLR 2025, PMLR 296:37–46.

Michał Daniluk and Jacek Dąbrowski. Temporal graph models fail to capture global temporal dynamics, 2023. URL https://arxiv.org/abs/2309.15730. arXiv version; TGL Workshop at NeurIPS 2023.

Abigail J. Hayes, Tobias Schumacher, and Markus Strohmaier. What do temporal graph learning models learn?, 2025. URL https://arxiv.org/abs/2510.09416v3. arXiv version 3.

Sture Holm. A simple sequentially rejective multiple test procedure. *Scandinavian Journal of Statistics*, 6(2):65–70, 1979. URL https://www.jstor.org/stable/4615733.

Zhenyu Hou, Haozhan Li, Yukuo Cen, Jie Tang, and Yuxiao Dong. GraphAlign: Pretraining one graph neural network on multiple graphs via feature alignment, 2024. URL https://arxiv.org/abs/2406.02953. arXiv preprint.

Shenyang Huang, Farimah Poursafaei, Jacob Danovitch, Matthias Fey, Weihua Hu, Emanuele Rossi, Jure Leskovec, Michael Bronstein, Guillaume Rabusseau, and Reihaneh Rabbany. Temporal graph benchmark for machine learning on temporal graphs. In *Advances in Neural Information Processing Systems 36*, NeurIPS 2023, pp. 2056–2073. Neural Information Processing Systems Foundation, Inc. (NeurIPS), 2023. doi: 10.52202/075280-0099. URL http://dx.doi.org/10.52202/075280-0099.

Xuanwen Huang, Wei Chow, Yize Zhu, Yang Wang, Ziwei Chai, Chunping Wang, Lei Chen, and Yang Yang. Enhancing Cross-domain link prediction via evolution process modeling. In *Proceedings of the ACM on Web Conference 2025*, WWW '25, pp. 2158–2171. ACM, April 2025. doi: 10.1145/3696410.3714792. URL http://dx.doi.org/10.1145/3696410.3714792.

Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. LightGBM: A highly efficient gradient boosting decision tree. In *Advances in Neural Information Processing Systems 30*, pp. 3146–3154, 2017. URL https://proceedings.neurips.cc/paper/2017/hash/6449f44a102fde848669bdd9eb6b76fa-Abstract.html.

Diederik P. Kingma and Jimmy Lei Ba. Adam: A method for stochastic optimization, 2014. URL https://arxiv.org/abs/1412.6980. arXiv version; published at ICLR 2015.

Emma Kondrup. Base3: a simple interpolation-based ensemble method for robust dynamic link prediction, 2025. URL https://arxiv.org/abs/2506.12764v2. arXiv version 2; TGL Workshop at KDD 2025.

Srijan Kumar, Xikun Zhang, and Jure Leskovec. Predicting dynamic embedding trajectory in temporal interaction networks. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '19, pp. 1269–1278. ACM, July 2019. doi: 10.1145/3292500.3330895. URL http://dx.doi.org/10.1145/3292500.3330895.

Yijun Ma, Zehong Wang, Weixiang Sun, and Yanfang Ye. Temporal graph pattern machine, 2026. URL https://arxiv.org/abs/2601.22454v3. arXiv version 3; GFM Workshop at ICML 2026.

Nguyen Minh Duc and Viet Cuong Ta. Sequence diffusion model for temporal link prediction in continuous-time dynamic graph, 2026. URL https://arxiv.org/abs/2601.23233. arXiv preprint.

Jiaxin Pan, Mojtaba Nayyeri, Osama Mohammed, Daniel Hernández, Rongchuan Zhang, Cheng Cheng, and Steffen Staab. Towards foundation model on temporal knowledge graph reasoning, 2025a. URL https://arxiv.org/abs/2506.06367v1. arXiv version 1.

Zhiqiang Pan, Chen Gao, Fei Cai, Wanyu Chen, Xin Zhang, Honghui Chen, and Yong Li. On the Cross-Graph transferability of dynamic link prediction. In *Proceedings of the ACM on Web Conference 2025*, WWW '25, pp. 4101–4110. ACM, April 2025b. doi: 10.1145/3696410.3714712. URL http://dx.doi.org/10.1145/3696410.3714712.

Farimah Poursafaei, Shenyang Huang, Kellin Pelrine, and Reihaneh Rabbany. Towards better evaluation for dynamic link prediction. In *Advances in Neural Information Processing Systems 35*, NeurIPS 2022, pp. 32928–32941. Neural Information Processing Systems Foundation, Inc. (NeurIPS), 2022. doi: 10.52202/068431-2386. URL http://dx.doi.org/10.52202/068431-2386.

Emanuele Rossi, Ben Chamberlain, Fabrizio Frasca, Davide Eynard, Federico Monti, and Michael Bronstein. Temporal graph networks for deep learning on dynamic graphs, 2020. URL https://arxiv.org/abs/2006.10637. arXiv version; ICML 2020 Workshop on Graph Representation Learning.

Kiarash Shamsi, Tran Gia Bao Ngo, Razieh Shirzadkhani, Shenyang Huang, Farimah Poursafaei, Poupak Azad, Reihaneh Rabbany, Baris Coskunuzer, Guillaume Rabusseau, and Cuneyt Gurcan Akcora. MiNT: Multi-network transfer benchmark for temporal graph learning. In *Advances in Neural Information Processing Systems 38*, NeurIPS 2025, pp. 150184–150218. Neural Information Processing Systems Foundation, Inc. (NeurIPS), 2025. doi: 10.52202/085713-4521. URL http://dx.doi.org/10.52202/085713-4521.

Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding, 2018. URL https://arxiv.org/abs/1807.03748. arXiv preprint.

Lu Yi, Jie Peng, Yanping Zheng, Fengran Mo, Zhewei Wei, Yuhang Ye, Zixuan Yue, and Zengfeng Huang. TGB-Seq benchmark: Challenging temporal GNNs with complex sequential dynamics, 2025. URL https://arxiv.org/abs/2502.02975. arXiv version; published at ICLR 2025.

Le Yu, Leilei Sun, Bowen Du, and Weifeng Lv. Towards better dynamic graph learning: New architecture and unified library. In *Advances in Neural Information Processing Systems 36*, NeurIPS 2023, pp. 67686–67700. Neural Information Processing Systems Foundation, Inc. (NeurIPS), 2023. doi: 10.52202/075280-2960. URL http://dx.doi.org/10.52202/075280-2960.

Zhenyu Yu, Chunlei Meng, Yangchen Zeng, Mohd Yamani Idna Idris, Jihong Guan, and Shuigeng Zhou. CaST-POI: Candidate-conditioned spatiotemporal ranking for next POI recommendation, 2026. URL https://arxiv.org/abs/2604.20845v2. arXiv version 2.

Xiaohui Zhang, Yanbo Wang, Xiyuan Wang, and Muhan Zhang. Efficient neural common neighbor for temporal graph link prediction, 2024. URL https://arxiv.org/abs/2406.07926. arXiv version; published at the Learning on Graphs Conference 2025.

Jianan Zhao, Zhaocheng Zhu, Mikhail Galkin, Hesham Mostafa, Michael Bronstein, and Jian Tang. Fully-inductive node classification on arbitrary graphs, 2024. URL https://arxiv.org/abs/2405.20445. arXiv version; published at ICLR 2025.

A Heuristic Bank and Model Details

This appendix specifies the 24 signals, their score transformations, and the primary query conditioner. Separately identified ablations vary the normalization or fusion rule.

Time and memory settings. The neighborhood capacity and source-gap history length are both $K = 20$. Windowed counts include events in $[t - W, t)$, with $W = 10^6$ in each dataset's unrescaled timestamp units. Exponential decay uses $\tau_{\text{decay}} = 10^6$ in the same units: an event at t_e contributes $\exp[-(t - t_e)/\tau_{\text{decay}}]$ to destination popularity. No cross-dataset conversion of timestamp units is applied. Table 4 separates the numeric unit from observation granularity. Where values are seconds, these settings give an 11.57-day window and e-folding time (8.02-day half-life); they do not equalize event rates across graphs. Wiki's physical unit is not explicit in the inspected upstream documentation, so we retain its provided cardinal time without claiming a verified cross-dataset unit conversion.

Table 4: Timestamp provenance for the frozen input versions. All local loaders preserve the provided numeric values. Endpoint samples describe observed granularity and do not certify full-stream resolution.

Dataset	Numeric-unit evidence	Observation granularity / qualification
Wiki v2	Relative cardinal timestamps; physical unit not explicit in JODIE/TGB instructions	Endpoint samples have native-unit gap GCD 1; this alone does not prove seconds
Review v2	Amazon 2018 `unixReviewTime`; TGB retains integer values	Endpoint samples lie on a daily lattice; the complete upstream reconstruction is not archived
GoogleLocal	Pinned TGB-Seq preprocessing directly reads `unixReviewTime` from the 2018 archive	Second-valued representation; endpoint-sample gap GCD 1, minimum positive gap 16
Coin v2	Local endpoints match the original ERC20 collection's GMT instants exactly as Unix seconds	Irregular observations with tied timestamps; stale `day` header does not define the unit
Flight v2	TGB explicitly uses Unix timestamps or `datetime.timestamp()`	Calendar-day observations; preprocessing timezone is not fully bound, so neither UTC midnight nor fixed 86,400-second spacing is assumed

The provenance record binds TGB commit `23349c3` and TGB-Seq commit `c1ee801`, and the GoogleLocal release `c818133`. It includes the original field conversions, registered input hashes and the sampled endpoints. The source accounts are the JODIE data-format instructions, Amazon 2018 data documentation, 2018 GoogleLocal documentation, and SNAP ERC20 collection record.

Direct neighborhood. For a node x, the direct neighborhood $N_K(x, t)$ is maintained as a ring buffer of capacity $K{=}20$ holding the most recent unique opposite-end neighbors that x has touched in events strictly before t. Each new event involving x refreshes the opposite endpoint's last-seen timestamp and moves it to the newest position if already present, or inserts it and evicts the oldest entry if the buffer is full; if x has touched fewer than 20 distinct neighbors so far, $N_K(x, t)$ simply holds all of them. The buffer is read and updated purely from history, so it never reflects the query being scored or any other event at the same timestamp. All four direct-structural channels in Table 5 (channels 12–15) are built from $N_K(u, t)$ and $N_K(v, t)$ for the two endpoints of a candidate pair.

Co-occurrence neighborhood. The co-occurrence neighborhood $N_K^{\text{co}}(v, t)$ is formed by following the K most recent incoming sources of v to their recent destinations, then retaining the K most recent unique destinations in this union. For opposite-side endpoints in a bipartite representation with disjoint node identities, as in Wiki, these destinations lie on the same side as u's direct neighbors. Their overlap with $N_K(u, t)$ can therefore be nonzero even though $N_K(u, t) \cap N_K(v, t)$ is empty.

Review's released shared-union ID representation does not satisfy this disjoint-role assumption (Appendix G.7). The candidate itself is excluded. A two-hop relation is dated by the later of its two edge timestamps; among multiple shared sources, we retain the latest such time. The 20 latest destinations are kept, breaking relation-time ties by node ID. Channels 16–19 measure overlap through common-neighbor, Jaccard, Adamic–Adar, and recency-weighted scores. For channels 15 and 19, each common neighbor contributes $\exp[-(t - \max(t_{uw}, t_{vw}))/\tau_{\text{decay}}]$, using the co-occurrence relation time in place of t_{vw} for channel 19.

The frozen source maps. Each channel is mapped into $[0, 1]$ by linear interpolation over 1,024 equally spaced empirical quantile levels. From 2,313,738 valid candidate rows in 110,178 source-training queries, we select the two million rows with the smallest full SHA256 identity hashes, using the same rows for all 24 channels. Each identity contains the dataset ID, global query row, source, positive destination, hexadecimal timestamp, and candidate destination; equal hashes are ordered by identity. Knots use linear quantile interpolation. This corrected fit restores the declared identity-based sampling rule; the preserved legacy fit instead used a seed-42 NumPy generator independently in channel order (Appendix C). Gap channels already contain their inverse-log transform; no additional log transform is applied. Values outside the fitted range are clipped to 0 or 1, and tied knots use their upper quantile level. Channels 12–15 have all-zero wiki training knots, so this rule maps their nonnegative inputs to 1 on every graph, including targets with nonzero direct overlap. These four channels are constant within each query and cannot themselves change its ranking in exact arithmetic. Their nonzero weights therefore do not measure ranking importance (Appendix H). We retain the same bank and maps for the primary and same-map fusion comparisons; normalization ablations use their own source training recipes. Logistic regression and LightGBM also use the corrected quantile map. BestSingle is the exception: it selects and scores one raw channel, cast to float64. No channel-removal ablation is reported.

Table 5 lists the complete frozen bank.

For both Adamic–Adar channels, $\deg_t(w)$ counts prior incoming and outgoing event incidences, including repeated interactions; a self-loop contributes twice. The implementation uses $\max\{\deg_t(w), 1\}$ in the denominator guard. Observed intersection members already have positive degree. This is neither a distinct-neighbor count nor a degree in the bounded co-occurrence graph.

Gap and context scores. Recency channels use $1/[1 + \log(1 + \Delta)]$ for elapsed time Δ. Source gaps are measured between distinct timestamps at which u was a source, retaining the last 20 gaps in G_u. Rhythm alignment is $1/[1 + |\Delta_u - \mu(G_u)|/(\sigma(G_u) + 1)]$, with population standard deviation, and requires at least two prior gaps. Channel 24 is $\log(1 + \text{median}(G_u)/(g_{\text{src}} + \epsilon))$, with $\epsilon = 10^{-12}$. The fixed source statistic $g_{\text{src}} = 300$ is the median of 103,709 consecutive per-event source gaps pooled from the original 110,232-event Wiki TRAIN prefix, retaining zero gaps at tied timestamps. It is reused unchanged on every target. The preserved legacy builders omitted this fit and used the constructor default $g_{\text{src}} = 0$. The correction does not alter native timestamp units or the causal history in G_u; Appendix C records the feature-repair boundary and provenance. Channel 23 uses the last 20 outgoing events; a destination counts as novel if it was globally unseen before that event's timestamp bucket. Channels 3, 4, 11, 23, and 24 are constant across a query's candidates but vary between queries, providing context to the conditioner.

The activity-memory score. For channel 20, each of the source's last 20 outgoing events within the window contributes its destination's decayed popularity, weighted by $\exp[-(t-t_e)/\tau_{\text{decay}}]$. Repeated destinations contribute once per retained event. We add the bounded lifetime pair count $c_{uv}/(1+c_{uv})$ to this sum and map the resulting nonnegative value z to $z/(1 + z)$. The pair-count term supplies its candidate dependence.

Missing history and event ordering. Absent history, empty structural intersections or unions, and undefined age or gap statistics produce zero raw scores. Rhythm alignment requires two source gaps; channel 24 requires one. These defaults precede quantile mapping, so a raw zero need not map to zero. All queries in a timestamp bucket are scored before that bucket updates history in canonical source–destination order.

Table 5: Frozen 24-channel heuristic bank (`dual_structural_v1`). N_K is the direct K-neighborhood, N_K^{co} the co-occurrence K-neighborhood (§A); $\deg_t(\cdot)$ is total prior incident event count. All channels are computed causally from state at time t using only events strictly before t.

#	Channel	Definition
Recency		
1	`pair_recency`	Inverse log-gap since the last (u, v) event
2	`destination_recency`	Inverse log-gap since the last event ending at v
3	`source_recency`	Inverse log-gap since the last event starting at u
4	`source_rhythm_alignment`	Alignment of the current source gap with u's recent gap mean/std
Frequency/Popularity		
5	`pair_count`	Lifetime count of prior (u, v) events
6	`windowed_pair_count`	Count of prior (u, v) events in window $[t - W, t)$
7	`repeat_indicator`	Indicator that (u, v) has occurred before
8	`destination_in_degree`	Lifetime prior in-degree of v
9	`destination_decayed_popularity`	Exponentially time-decayed popularity of v
10	`windowed_destination_popularity`	Windowed prior in-degree of v
11	`source_activity_count`	Lifetime prior out-activity count of u
Structural (direct)		
12	`direct_common_neighbors`	$\lvert N_K(u,t) \cap N_K(v,t)\rvert$
13	`direct_jaccard`	Jaccard index of $N_K(u,t)$, $N_K(v,t)$
14	`direct_adamic_adar`	$\sum_{w \in N_K(u,t) \cap N_K(v,t)} 1/\log(1+\deg_t(w))$
15	`direct_recency_weighted_cn`	Direct common neighbors, exponentially recency-weighted
Structural (co-occurrence)		
16	`cooccurrence_common_neighbors`	$\lvert N_K(u,t) \cap N_K^{\text{co}}(v,t)\rvert$
17	`cooccurrence_jaccard`	Jaccard index of $N_K(u,t)$, $N_K^{\text{co}}(v,t)$
18	`cooccurrence_adamic_adar`	$\sum_{w \in N_K(u,t) \cap N_K^{\text{co}}(v,t)} 1/\log(1+\deg_t(w))$
19	`cooccurrence_recency_weighted_cn`	Co-occurrence common neighbors, recency-weighted
Structural (context)		
20	`tcomem`	Bounded recent-destination activity and pair memory
21	`preferential_attachment`	$\log(1+\text{outdeg}_t(u)) \cdot \log(1+\text{indeg}_t(v))$
Context/Novelty		
22	`node_age_geometric_mean`	Geometric mean of u's and v's log-ages
23	`source_novelty_rate`	Fraction of u's recent events reaching a globally unseen destination
24	`source_local_timescale`	Log-scaled median of u's recent inter-event gaps

A.1 Exact query summaries and network

Let $\widetilde{x}_{ik}$ be the normalized score of valid candidate i on channel k, with $N = Q + 1$ valid candidates. The conditioner receives five statistics for each of 24 channels:

$$z_k = \left(\mu_k, \sigma_k, m_k, \frac{\exp(H(p_{\cdot k}))}{N}, g_k\right), \qquad p_{ik} = \text{softmax}_i(\widetilde{x}_{ik}). \tag{2}$$

Here μ_k, σ_k, and m_k are the mean, population standard deviation, and maximum of the normalized scores. Entropy H is in nats. The gap g_k is the maximum minus the second-largest distinct score, or zero when all scores tie. The positive occupies index zero; valid positions satisfy $0 \le i \le Q(q)$, where the stored `neg_count` is $Q(q)$. Padding is excluded from summaries, loss, and ranking, and receives score $-\infty$. The entropy statistic is an effective candidate fraction: it equals one for equal scores and decreases as score mass becomes more concentrated. The training-loss temperature is

not used in these summaries. For an equally sized mixture of zeros and ones, the effective candidate fraction is $\frac{1+e}{2}\exp[-e/(1+e)] \approx 0.894989$, whereas equal scores give one. Its sensitivity depends on the score distribution; one isolated high score is not representative of its full range.

Duplicating every valid candidate equally preserves all five statistics, up to floating-point rounding. This is the candidate-count invariance used in the implementation; resampling a different candidate population can change them. This summary invariance does not imply that InfoNCE is invariant to the number or distribution of negatives: its denominator and score-scale dependence remain. The primary receives no raw count or $\log(Q+1)$ input. The `+logQ` ablation adds that scalar, while `candidate_independent` supplies zeros in place of the summaries to learn a single global weight vector.

The concatenated summaries form a 120-dimensional input to the network $\text{Linear}(120, 128) \rightarrow \text{ReLU} \rightarrow \text{Linear}(128, 24)$. Its softmax output gives the 24 channel weights in Eq. 1. The optional shared calibration network is disabled in every reported run, leaving 18,584 parameters. Source training uses listwise InfoNCE with temperature $T_{\text{loss}} = 0.01$ and Adam at learning rate 3×10^{-4}. The complete frozen recipe is `tgany_dev_v3`, specified with its source-selection history in Appendix I.6.

A.2 Streaming computation and transfer

All reported computation runs on CPU. First, compute the bank in chronological order and fit quantile maps to source training outputs. Second, train the mixture on source queries and select checkpoints on `dev_tune`. Third, reuse the learned parameters and maps on each target, computing fresh summaries and weights from its causal history. Score every query at a timestamp before updating the state with that timestamp's events. At fixed neighborhood capacity and history windows, state maintenance is an $O(E)$ sweep; candidate scoring additionally depends on the number of queries and candidates. Corrected same-feature evaluation uses one numerical thread and fixed 64-query blocks, each retaining the complete candidate axis and original query order. The 64-query block size is a fixed execution policy; no cross-batch bitwise or rank-invariance claim is made. Reciprocal ranks use the exact average-tie rule in Appendix B.4.

Specified target-calibration extension. The separately defined `target_calibrated` regime would refit quantile maps on target pre-test history (training and validation events), without ranking labels, while keeping the model parameters frozen. This is target adaptation. It has not been evaluated; all reported TG-Any transfer evaluations use `strict_zs`, with source-fitted maps.

B Preregistered evaluation protocol

This appendix specifies the data splits, controls, negative sampling, and statistical analysis behind §4.1. In the experiment records, `strict_zs` denotes source-trained zero-shot transfer, K2 denotes the six-control source-development comparison, and P2 denotes the original official-test session. Appendix D describes the separate Base3 reproduction study. This section preserves those historical protocol identities. The versioned corrected campaign is a post-hoc rescore after outcomes were inspected; it does not reopen the K2 gate or subsequent test-entry decisions, and it does not turn corrected official-set scores into untouched confirmation (Appendix C).

B.1 Evaluation conventions

The experiment plan fixed the following conventions before official testing.

1. **Separate development from official testing.** Test data could be downloaded, hashed, and checked for schema and split counts. Test metrics were reserved for a separately authorized evaluation after freezing the code, environment, data, sampler, features, hyperparameters, controls, seeds, statistical analysis, and decision thresholds.
2. **Distinguish transfer regimes.** `strict_zs` uses source-fitted transforms and weights. `target_calibrated` permits label-free target pre-test history to recalibrate transforms while keeping weights fixed; it is unsupervised target adaptation (Appendix A.2).

3. **Keep candidates and event order consistent.** Training and pseudo-development use the deterministic sampler below. Official evaluation uses the supplied negatives unchanged. Every streaming evaluation scores events before updating state (§3.1).

4. **Fix the comparison and representation.** K2 compares with the strongest of six controls, including three same-feature controls (§4.2). Its criterion requires TG-Any's mean to exceed all six, with at least four of five seeds exceeding the strongest control. The `dual_structural_v1` bank contains both direct and co-occurrence channels on every graph (§3.2).

5. **Account for temporal dependence.** Confidence intervals use the paired temporal block bootstrap in §B.7, rather than an independent-query bootstrap.

B.2 Development and official testing

The planned sequence was to fit on source training data, select hyperparameters on source validation, evaluate development comparisons and sensitivity, then freeze the configuration before official testing. The source, `tgbl-wiki`, supplied all TG-Any training data and feature-normalization fits. Its validation split was divided chronologically into `dev_tune` (11,810 queries) and `dev_gate` (11,811 queries). The timestamp boundary minimizes its distance from an exact half split, with ties resolved toward the earlier boundary. Hyperparameters were selected on `dev_tune`; `tgbl-review` validation served as the held-out development transfer target. The planned GoogleLocal pseudo-development evaluation was not run.

Engineering checks, the six-control K2 comparison, and source candidate-count sensitivity were completed before the freeze. The planned uniform- and popularity-sampler comparisons remain unrun (§B.6). The freeze recorded code and environment versions, data and sampler hashes, features, hyperparameters, controls, seeds, statistical code, and decision thresholds.

The original P2 session and its GoogleLocal continuation produced official results for wiki, review, and GoogleLocal. Coin's official test belonged to the later expansion (Appendix F). Each original primary test evaluation ran once. Later post-hoc Uniform/Global comparisons score additional predictors on frozen target caches and reuse the original primary RR vectors (Appendix G); they are not new independent confirmation. The protocol requires any later bug-fix evaluation to receive a separate authorization and version, preserving the original record. The corrected campaign has that separate version: it refits affected source-trained predictors and source- or target-window controls, then rescores matched methods under a common corrected feature, map, and runtime contract. Because target outcomes were already known, these rescores remain post-hoc; they do not replace the historical gate decision, repeat the original one-time test claim, or authorize a Flight official-test evaluation.

B.3 P2: the official-test session

Run `20260724T_p2_confirmatory` used 6 SHA-256-addressed manifests covering the experiment plan, code, dataset snapshot, sampler, baselines, and statistical plan, with the seed list $\{0, 1, 2, 3, 4\}$.

Before opening official test data, the runner verified all 90 frozen files, including the five checkpoints, configurations, normalizer, and code. The saved session record links these inputs to the completed evaluations.

Evaluator parity was checked using a representative BestSingle score grid for each dataset, against `py-tgb` 2.2.0 for Wiki and Review and `tgb-seq` 0.1.2 for GoogleLocal; the checks did not cover every model cell separately (Appendix E.3).

The session completed wiki and review but encountered a GoogleLocal loader error during raw-CSV namespace validation, before scoring any GoogleLocal query. After a loader repair checked before scoring, GoogleLocal was evaluated in run `20260724T_p2_confirmatory_v2_googlelocal`. Appendix I.7 records the historical repair and receipt finalization. Table 1 now reports the separately versioned corrected score on the same official split.

B.4 Training and pseudo-development negative sampling

Training uses `NEG-TRAIN-v1`, the deterministic `causal_hist_rnd_v1` sampler with master seed 42 and $Q{=}20$ negative destinations per query. Eligible destinations must have appeared strictly before the query timestamp. All destinations positive for the source at that timestamp are excluded.

Half the negatives are sampled without replacement from the source's historical destinations. The other half come from the observed nonhistorical destinations, with cross-filling when either pool is too small. Candidates are unique. Queries with fewer than Q eligible negatives are skipped and logged. Sampler outputs are content-addressed for reproducibility.

GoogleLocal has no official validation negatives. Its unrun pseudo-development protocol uses `NEG-DEV-v1`: $Q{=}100$ with the same causal and exclusion rules and a separate manifest. The planned `uniform_v1` and `popularity_075_v1` comparisons at $Q \in \{20, 100, \text{full}\}$ were not completed.

Official validation supplies 999 negatives per query on `tgbl-wiki-v2` (998 for two queries) and 100 on `tgbl-review-v2`. GoogleLocal supplies 100 official negatives per *test* query. Its downloaded negative file contains duplicate destinations in 741 of 43,546 rows, with at least 98 unique destinations per row. We retained these supplied rows unchanged.

For positive score s^+ and valid negative-position scores s_j^-, the benchmark's average optimistic/pessimistic rank and reciprocal rank are

$$r = 1 + \frac{\#\{j : s_j^- > s^+\} + \#\{j : s_j^- \geq s^+\}}{2}, \qquad \text{RR} = 1/r.$$

Duplicate destinations retain their supplied positions in both counts; padding is excluded. No destination deduplication precedes ranking. The corrected evaluator applies this rule in float64 with no tie tolerance and rejects nonfinite scores at valid candidate positions.

B.5 Baseline and control configurations

Logistic regression uses the 24-channel quantile-normalized bank, $C{=}1$, the `lbfgs` solver, ℓ_2 regularization, and seed 42. LightGBM uses the same bank, 31 leaves, 400 estimators, learning rate 0.05, and deterministic seed 42.

For wiki, review, and GoogleLocal, both fitted controls are trained on wiki shards and scored with the wiki normalizer. BestSingle instead selects one raw channel on all 11,810 wiki `dev_tune` queries and scores that raw channel in float64. Each original confirmatory session fits its controls deterministically from those fixed source inputs before scoring, rather than loading serialized control checkpoints. Corrected controls are refit from the corrected, hash-bound source inputs before their post-hoc scoring. The coin and flight evaluations instead fit controls on target training windows (Appendix F); their raw BestSingle channel is selected on the fixed first 50,000 validation queries, so its full-validation score overlaps selection as disclosed in Appendix C.

Published TGN (Rossi et al., 2020), DyGFormer (Yu et al., 2023), and JODIE (Kumar et al., 2019) results provide context where available. These models learn representations from the event stream; the same-feature comparisons evaluate alternative predictors over the fixed bank.

B.6 Candidate-count and sampler sensitivity

The planned sensitivity study covered $Q \in \{20, 100, \text{full official set}\}$ under three negative samplers. The completed study varies candidate count on source `dev_gate` (Table 17). Additional samplers and transfer-target candidate-count sweeps remain untested; each expansion target uses one candidate width.

The `candidate_independent` ablation removes the within-set distributional inputs while retaining the bank and training objective. It initially tests query conditioning on source and Review validation; the same frozen weights were later reused on Coin and Flight validation (§E.5).

B.7 Statistical analysis

Corrected confidence intervals use paired per-query reciprocal-rank differences joined by exact query identity, timestamp, and candidate count. TG-Any's reciprocal ranks are first averaged within query over the fitted seeds prescribed for that comparison: five for the primary analyses and matched seeds 0–2 for the three-seed supplementary comparisons. Complete timestamp groups are then packed into contiguous blocks at factors 0.5, 1, and 2 of $B = \max(1000, \lceil n_{\text{queries}}/100 \rceil)$, without splitting a timestamp group. We use $10{,}000$ bootstrap resamples and weight sampled blocks by their query counts. Each interval is pointwise at its stated block length and describes query-block variability conditional on those fitted models; it is not a simultaneous interval over the three lengths. Across-seed standard deviations are reported separately. Three-seed results are labelled exploratory. LR and LightGBM use fixed seed 42, and the non-neural rule-based controls are deterministic. The corrected Review and expansion validation intervals use the implementation described in Appendix G.4; old row-index caller outputs are preserved as historical records.

The historical routine also calculates twice the smaller bootstrap sign-tail fraction, capped at 1, and applies Holm's procedure within each declared family (Holm, 1979). Because these resamples are centered on the empirical effect rather than generated under a null hypothesis, these quantities are not calibrated hypothesis-test p-values; Holm adjustment alone does not make them so. For the headline positive comparisons, none of the $10{,}000$ resampled differences was nonpositive. The stored raw and adjusted estimates are consequently zero. We report tail counts with effect sizes and intervals; zero counts do not establish raw or adjusted $p < 10^{-4}$. We make no new significance or family-rejection claims from these historical tail estimates.

K2 uses the preregistered mean-advantage and seed-win criterion. The expansion uses its separately specified mean inequality. Both are historical decision rules; the corrected post-hoc analyses do not retroactively rerun either gate.

C Post-hoc pipeline correction and evidence boundaries

The corrective campaign re-evaluates previously inspected datasets after an implementation and statistical audit. It retains the original source graph, architecture, training recipe, evaluation sets and prescribed seeds. These results are post-hoc corrections, not new untouched confirmation. The original source gate and the coin/flight test-entry decisions remain historical decisions; corrective scores do not reopen those decisions or authorize a flight official-test evaluation.

Feature construction and normalization. The historical feature builder used a default value of zero for the source-gap reference in channel 24. The correction fits the existing median-gap definition on wiki TRAIN only: 110,232 events yield 103,709 consecutive per-source gaps, including five zero gaps, and a reference of 300 in the original timestamp units. That same scalar is used on every graph. Each query's past-gap state is reconstructed from causal raw history, and only valid-candidate channel 24 is replaced. The remaining 23 channels, query identities, timestamps, candidates, padding and order are preserved. Timestamp units are not adjusted using target outcomes. The normalization sample follows the declared stable-identity SHA256 rule: a common sample of 2,000,000 of the 2,313,738 valid source TRAIN candidate rows fits the 24 quantile maps. The historical per-channel seeded random sample was source-only but differed from this declared rule. This discrepancy does not establish target leakage or explain a performance deficit on its own.

Execution and matched comparisons. Corrected models and fitted controls bind their configuration, normalizer, channel order, input manifests, implementation and fitted parameters to exact file hashes. Valid nonfinite features or scores raise an error. These checks address demonstrated failure modes; they do not establish that historical reported runs encountered nonfinite values. Corrected same-feature scoring uses Python 3.10.20, NumPy 2.0.2 and Torch 2.5.1 on CPU with one numerical thread and batches of 64 complete queries. Learned and fixed-weight scores retain float32, while fitted controls retain their native float64 scores; reciprocal ranks use float64 and the exact average-tie rule without a tolerance. Each compared method reads the same corrected candidate features. Historical primary reciprocal ranks and historical-map controls are not substituted into corrected matched comparisons. The separately frozen CrossDyG author implementation retains its own four-thread,

Table 6: Post-hoc pipeline correction on the same query cohorts. MRR is averaged over the five fixed seeds. Differences and pointwise 95% temporal-block intervals are conditional on these fitted seeds; the combined correction does not isolate one repair. Official test split labels identify the datasets, not a new untouched confirmation.

Cohort	Legacy	Corrected	Δ	95% interval
Wiki gate	0.8421	0.8421	-0.000028	[-0.000138, +0.000069]
Wiki gate ($Q = 20$)	0.8765	0.8765	+0.000002	[-0.000154, +0.000155]
Wiki gate ($Q = 100$)	0.8576	0.8576	-0.000069	[-0.000159, +0.000018]
Wiki test	0.8290	0.8290	+0.000004	[-0.000090, +0.000094]
Review validation	0.1365	0.1366	+0.000050	[+0.000043, +0.000056]
Review test	0.1613	0.1614	+0.000093	[+0.000083, +0.000104]
GoogleLocal test	0.2380	0.2379	-0.000187	[-0.000248, -0.000127]
Coin validation	0.7857	0.7851	-0.000684	[-0.000718, -0.000649]
Coin test	0.8068	0.8057	-0.001034	[-0.001064, -0.001003]
Flight validation	0.8900	0.8895	-0.000474	[-0.000506, -0.000443]

per-query execution policy. Its three-seed full Review validation run completed under the original fixed budget; the corrected paired comparison is reported in Appendix G.3.

Graph rules and target selection. The canonical wiki LocalGlobal evaluation removes an older unseen-positive zero-rank override: exactly 32 of 23,621 validation queries change, including 13 of 11,811 source-gate queries; the other 23,589 reciprocal ranks are bit-identical. Original reference-anchor and gate records are retained as history. Target BestSingle continues to select its channel on the first 50,000 validation queries, so its full-validation score overlaps selection. A separate sensitivity excludes the complete timestamp bucket containing the last selection row and all earlier rows, retaining 3,371,403 coin and 10,040,574 flight queries. This fixed suffix keeps the already selected channel and is not an untouched test or a replacement for the full evaluation matrix.

Statistical interpretation. Corrected paired analyses join exact query identifiers and verify real timestamps and candidate counts. They average the prescribed seeds within each query before resampling complete timestamp groups, using three block lengths and query-count weighting. Seeds are not treated as independent query observations. Intervals are pointwise and conditional on the fitted seeds; descriptive bootstrap sign-tail fractions are not calibrated null-test p-values. Old-versus-corrected primary differences combine preprocessing and numerical-execution changes and cannot identify the isolated causal effect of one repair. The historical six-control K2 decision and intervals remain explicitly historical; current paired claims are limited to comparators with independently verified complete reciprocal-rank evidence.

Complete matrix and preserved legacy comparison. All ten fixed cohorts and 375 scoring shards completed under the corrected recipe, followed by the prescribed statistics, diagnostics, and selection suffix analyses. Table 6 compares all five legacy and corrected primary seeds on each identical query cohort. Primary changes are small and not uniformly favorable: for example, Review official changes by +0.000093 MRR, Coin official by -0.001034, and Flight validation by -0.000474. These combined changes do not identify the causal effect of an individual repair. They also do not measure changes in refitted controls, which must be read from their own corrected matched comparisons. The separate CrossDyG queue and final package assembly are not certified by completion of this matrix.

D Reproducing the Base3 baseline

In the historical baseline study, two independently built Base3 implementations produced the same wiki validation MRR, below the published value. This appendix describes the comparison, the implementation checks, and the remaining reproducibility gap behind §4.1 and §E.1. These are preserved original runs, not corrected-model comparisons. The historical LocalGlobal result contains the old unseen-positive zero-rank override; the canonical correction is reported separately in Appendix C.

D.1 Baseline reproduction check

We implemented `Base3-paper-clean-v1` from the published method description, using reference-compatible batches of 200 events. The baseline reproduction run (P1-A, `20260711T212309Z`) evaluated it alongside EdgeBank and LocalGlobal (Cornell et al., 2025) on `tgbl-wiki` validation. Code and environment were frozen and independently reviewed before execution. Table 7 reports the recorded results (receipt SHA-256 `720693f3a932…`), with observed MRRs reported to six decimal places and validation anchors to three decimal places.

Table 7: Baseline reproduction on `tgbl-wiki-v2` validation, run `20260711T212309Z`.

Baseline	Observed MRR	Validation anchor
EdgeBank (fixed time window)	0.599741	0.600
EdgeBank (unlimited)	0.526654	0.527
LocalGlobal	0.841527	0.842
Base3 (`Base3-paper-clean-v1`)	0.704373	0.727

EdgeBank and LocalGlobal matched their published anchors within the preregistered absolute tolerance of 0.002. Base3 reached 0.704373, a gap of 0.022627 from 0.727. We investigated the discrepancy before using Base3 in the development comparison. The LocalGlobal anchor is the validation entry in the official TGB leaderboard; its published test score is 0.821.

The frozen configuration matched the published wiki setting: memory span 0.1, co-occurrence weight 1.0, PopTrack $K{=}1000$, and `multi_conf` interpolation. This is Base3's rule-based, candidate-dependent weighting scheme. The public commit's runnable scripts, however, hard-code review- and coin-specific negative files. No author-provided wiki command, negative-file hash, environment, or output receipt was available for comparison.

The initial diagnosis identified several possible implementation sensitivities: dense float32 versus sparse float64 PopTrack state, NumPy-dependent top-K ordering and ties, and sequential arithmetic. The next experiment tested whether reproducing these source-level semantics changed the result.

D.2 Source-compatibility checks

We specified `Base3-source-compat-v2` before accessing validation negatives. The specification fixed memory span, co-occurrence weight, PopTrack K, dense float32 state, full node-universe construction, sequential in-place arithmetic, and subcomponent update order. It also fixed the top-K ordering and tie behavior of NumPy `1.23.5`. The pinned public source served as a behavioral reference; its code was not copied, adapted, or redistributed.

Verification had four stages:

1. **Synthetic differential tests.** An independent oracle covered ten edge-case categories: zero-popularity ties; K beyond the observed-destination count; long-stream float32 accumulation; all four interpolation branches; quantile boundaries; batch-of-200 score-then-update behavior; ring-buffer fill, overflow, and eviction; time-window endpoints; symmetric co-occurrence updates; and full t-CoMem score composition. Expected states, tie rules, and tolerances were fixed in advance across suites of 12 and 8 cases. Failed checks required a results-blind repair and a newly hashed test run.
2. **Real-stream state trace.** After synthetic verification, we compared v2 and oracle states at fixed positions in wiki training and validation-positive streams. This trace used no validation negatives and computed no performance metric. It covered EdgeBank membership, PopTrack arrays and ordering, t-CoMem deques and scores, and score-then-update boundaries.
3. **Independent review.** A reviewer who did not implement v2 checked provenance, hashes, test and trace outputs, runner access controls, and official-test isolation.
4. **One validation evaluation.** We then evaluated v2 once with the unchanged official validation negatives and evaluator. Agreement with the published MRR was not an acceptance criterion; the result was reported without tuning or rerunning against it.

D.3 Verification and validation results

Both synthetic suites passed, 12/12 and 8/8, in the pinned Linux/NumPy `1.23.5` environment. The real-stream trace covered 119 batches, 23,621 validation events, and 25,263 candidates. All 101,052 compared component values matched, with zero component, state-byte, or state-map differences. Access logs recorded one approved data open, no validation-negative or forbidden-artifact accesses, and no evaluator import. Appendix I provides the runtime attestation and pre-metric review record.

Run `20260723T_v2_descriptive_r3` obtained 0.7043730536099085 over 23,621 wiki validation queries. We retain full double precision here because the result exactly matches `Base3-paper-clean-v1`'s independently frozen P1-A value. The two implementations therefore agree despite being built from different references: the method description and the pinned source's operational semantics.

The aggregate absolute difference from the official `py-tgb 2.0.0` evaluator was 4.42×10^{-10}. That version matches the Base3 environment; the P1/P2 sessions used `py-tgb 2.2.0` (§B.3). Evaluator arithmetic and the tested state, ordering, and precision choices do not explain the gap to 0.727. Reproducing that published value remains unresolved without a complete wiki execution record. The historical comparisons therefore identify both reimplementations explicitly and retain the published result as external context.

E Additional Results and Experimental Details

This appendix collects dataset sizes, source and transfer comparisons, seed detail, ablations, and candidate-count sensitivity. Current results use the corrected pipeline; the original six-control gate and exploratory three-seed development table are explicitly historical. Results use validation data except where labelled official splits.

Table 8: Dataset sizes from the frozen registry. GoogleLocal's node count sums its disjoint source and destination namespaces; Review's count uses the released shared-union IDs (Appendix G.7). Wiki supplies training data; review supplies development transfer. GoogleLocal pseudo-development was not run. All three have separate official tests.

Dataset	Nodes	Edges	Train	Val.	Test
`tgbl-wiki-v2`	9,227	157,474	110,232	23,621	23,621
`tgbl-review-v2`	352,637	4,873,540	3,413,837	730,784	728,919
`GoogleLocal`	473,580	1,913,967	1,802,833	67,588	43,546

E.1 Historical source gate and corrected source comparison

Table 9: Historical six-control K2 gate before pipeline correction on wiki (`dev_gate`, n=11,811). TG-Any reports a five-seed mean ± std; controls are deterministic. All five TG-Any seeds exceed the strongest control (LocalGlobal).

Method	`dev_gate` MRR
LocalGlobal	0.8357
`Base3-paper-clean-v1`	0.6976
`Base3-source-compat-v2`	0.6976
Best single channel (`pair_recency`)	0.8331
Logistic regression (24-ch. bank)	0.8291
LightGBM (24-ch. bank)	0.5433
TG-Any	0.8421 ± 0.0003

The table preserves the original primary, controls, ranking convention, and gate decision. In the corrected source-gate comparison, TG-Any reaches 0.8421; Global reaches 0.8383 and canonical LocalGlobal 0.835699. TG-Any exceeds the latter by 0.006387 (base 95% interval $[0.004369, 0.008446]$),

with all five seeds ahead. The corrected LocalGlobal comparison removes the old unseen-positive zero-rank override. Base3 was not rerun in the corrected matrix, so these current results do not recreate a six-control K2 family.

On wiki's official test, TG-Any reaches 0.8290 and continues to exceed every same-feature control in Table 10. The published LocalGlobal result is 0.821 (Cornell et al., 2025), as external test-set context. It is distinct from the leaderboard validation anchor near 0.842 and from our matched, canonical validation comparison. Subtracting a published rounded test score is not a matched paired comparison. Our Base3 comparison uses two reimplementations whose results agree at 0.704373 on full wiki validation (23,621 queries), rather than the 11,811-query gate used in Table 9, compared with the paper's reported 0.727. Appendix D documents that discrepancy and the implementation checks behind the comparison.

Table 10: Corrected Wiki official-set comparison (n=23,621). All predictors use the same fixed bank. Learned models use source-fitted maps; Best single channel uses raw values. TG-Any reports five-seed mean $\pm$ std; the other rows are deterministic.

Predictor	Wiki official-test MRR
Best single channel	0.8172
Logistic regression	0.8165
LightGBM	0.6480
TG-Any	0.8290 $\pm$ 0.0002

E.2 Historical six-control bootstrap record

Table 11 preserves the original six comparisons underlying K2 (§E.1). We used 10,000 paired temporal block resamples, block length $\max(1000, \lceil n_{\text{queries}}/100 \rceil) = 1{,}000$, a fixed seed, and pointwise percentile intervals. The historical record also applied Holm to bootstrap sign-tail fractions; these are not calibrated null-test p-values and we do not interpret them as familywise error control. The intervals condition on the five fitted TG-Any models (§B.7).

Table 11: Historical paired temporal block bootstrap on `dev_gate`: TG-Any minus each K2 control, in absolute MRR. The last column counts nonpositive differences among $10{,}000$ resamples. Inference is conditional on the five fitted TG-Any models (§B.7).

Comparison	ΔMRR	95% CI	$\Delta^* \leq 0$ / 10,000
TG-Any vs. LocalGlobal	0.0064	[0.0044, 0.0085]	0
TG-Any vs. Best single channel	0.0090	[0.0066, 0.0115]	0
TG-Any vs. Logistic regression	0.0130	[0.0105, 0.0156]	0
TG-Any vs. `Base3-paper-clean-v1`	0.1446	[0.1244, 0.1650]	0
TG-Any vs. `Base3-source-compat-v2`	0.1446	[0.1244, 0.1650]	0
TG-Any vs. LightGBM	0.2988	[0.2872, 0.3112]	0

Against LocalGlobal, the strongest control, TG-Any's mean advantage was 0.0064, with its 95% interval above zero. All five seeds exceeded the anchor, satisfying the preregistered K2 criterion. The stored raw and Holm-adjusted tail estimates are zero; their finite resampling resolution is explained in §B.7. These original estimates are not replaced with corrected-model means. The corrected same-feature and canonical LocalGlobal contrasts have separate complete query-paired reports; there is no corrected Base3 contrast or retrospective K2 decision.

E.3 Official-test paired differences

Table 12 gives the official-test advantage over the strongest of the three original source-trained same-feature controls on each dataset. These paired temporal block intervals condition on the five fitted TG-Any models, using the procedure in §B.7.

Table 12: Corrected post-hoc official-set MRR differences between TG-Any and the strongest of the three original source-trained same-feature controls.

Dataset	Control	ΔMRR	95% CI
Wiki	Best single channel	+0.0117	[+0.0107, +0.0129]
Review	Logistic regression	+0.1352	[+0.1329, +0.1374]
GoogleLocal	LightGBM	+0.0456	[+0.0429, +0.0482]

Historical official evaluator checks. Each original confirmatory runner checked a representative best-single-channel score grid, rather than every model cell, against its official evaluator. Maximum absolute errors were 1.17×10^{-8} on wiki, 2.19×10^{-9} on review, and 6.8×10^{-10} on GoogleLocal (`tgb-seq 0.1.2`). All were within 5×10^{-7}. These checks describe the original runner; corrected scoring has its own hash-bound exact-tie and input validation contract (Appendix C).

E.4 Per-seed results

TG-Any's source `dev_gate` MRRs for seeds 0 through 4 were 0.842333, 0.841785, 0.842417, 0.842158, and 0.841736. All exceed the corrected canonical LocalGlobal mean of 0.835699. This current five-seed comparison is separate from the original gate in Table 9.

The corrected scores on the three original official splits also use five-seed means. Their per-seed ranges were [0.828754, 0.829186] on wiki, [0.160011, 0.162298] on review, and [0.234635, 0.242975] on GoogleLocal (Tables 10 and 1).

E.5 Uniform and linear mixture controls

The corrected supplementary analyses use the corrected source and target feature views, refitted Wiki quantile maps, and fixed candidate masks and exact tie-aware reciprocal ranks. Uniform assigns weight $1/24$ to every normalized channel. It has no learned weights. The linear ranker has 24 unconstrained coefficients, no bias, and no query conditioner. It uses TG-Any's listwise InfoNCE objective with temperature 0.01 and Adam with learning rate 0.0003, betas $(0.9, 0.999)$, epsilon 10^{-8}, and no weight decay. Training uses batches of 256 for at most 30 epochs, with patience 5. Seeds 0–4 select epochs 9, 6, 16, 12, and 10 using wiki `dev_tune` alone. All five selected checkpoints are evaluated. The global mixture retains TG-Any's nonnegative, sum-to-one weights; the linear ranker allows signed coefficients. Both use the same ranking objective, making the parameterization a substantive difference.

The original supplementary plan first covered Wiki gate and Review validation, then added Coin and Flight after the Review ordering was known. The corrected campaign preserves that post-hoc status and refits all 29 prescribed learned models, including the five linear seeds, before complete fixed-matrix scoring. No target outcome selects a new fusion rule or replaces the declared primary. The original fit and score records remain archived; they are not mixed into the corrected matched comparison.

The linear control reaches 0.8257 ± 0.0019 on the source, but only 0.0526 ± 0.0316 on review. Its review MRR ranges from 0.0246 to 0.0957 across seeds. Uniform has much lower source MRR (0.3522) and much higher review MRR (0.2678). Thus source ranking accuracy alone does not predict transfer through this feature bank.

Corrected scoring binds the numerical environment, native score dtypes, 64-query complete-candidate batches, and exact average-tie rule. Cross-implementation or cross-batch equivalence is not assumed: float32 reduction changes can affect ties. The stored complete reciprocal ranks, query identities, and small reports are checked under the prescribed runtime; independent summaries do not rescore models.

E.6 Conditioning details

Matched normalization transfer is reported separately in Appendix G.5.

Table 13: Conditioning ablations on wiki held-out validation and review validation. Every row uses five seeds and reports mean ± std. `+logQ` adds $\log(Q+1)$ to the existing summaries; it does not remove their candidate-distribution information.

Block / Variant	Wiki val. MRR	Review val. MRR	Seeds
How are mixture weights chosen?			
Query-conditioned (ours)	0.8421 ± 0.0003	0.1366 ± 0.0005	5
`+logQ`	0.8420 ± 0.0006	0.1361 ± 0.0022	5
Global learned mixture	0.8383 ± 0.0002	0.1619 ± 0.0071	5

E.7 Three-seed conditioner ablations

Table 14 retains the original exploratory three-seed development results as history. All its values precede the pipeline correction. The current matched five-seed Wiki and Review comparisons are in Table 13; the historical tune, gate, and transfer values below are not their current counterparts.

Table 14: Historical exploratory conditioner ablations using seeds $\{0, 1, 2\}$ (`exploratory_three_seed`). All columns use the same three seeds. This table retains the original three-seed record; corrected five-seed results appear in Table 13.

Variant	Wiki tune	Wiki gate	Review val.
`+logQ`	0.8525	0.8423	0.1365
Global weights	0.8453	0.8383	0.1627

E.8 Development transfer to tgbl-review

Table 15 evaluates the frozen wiki weights and normalizer on review's official validation split. This split served as a held-out development target, with no target fitting, calibration, or parameter selection.

Table 15: Zero-shot transfer to `tgbl-review` official validation (n=730,784), the development transfer target. TG-Any is a five-seed mean ± std. All predictors are trained or selected on Wiki. TG-Any, logistic regression, and LightGBM use the Wiki-fitted normalizer; BestSingle ranks the selected raw channel in `float64`.

Method	`tgbl-review` val. MRR
Best single channel (`pair_recency`)	0.0193
LightGBM (24-ch. bank)	0.0234
Logistic regression (24-ch. bank)	0.0366
TG-Any (strict_zs)	**0.1366 ± 0.0005**

Published test results provide separate context: EdgeBank-tw ∼0.025 and TGN ∼0.349 (Huang et al., 2023). TGN is trained on review; these test results are not matched to the source-trained validation comparison above.

Each same-feature control had MRR below 0.04. TG-Any reached roughly $3.7\times$ logistic regression and $7.1\times$ the best single channel. The learned mixture therefore transferred substantially better than these controls. Global weights transferred even better, reaching 0.1619 on the same split (§5.3).

E.9 The gain extends to previously unseen pairs

We label a validation positive as repeated when its source–destination pair occurred strictly before the query timestamp. Labels come from the raw positive candidate's `repeat_indicator`, cross-

checked against `pair_count>0`. Both use the feature builder's complete-timestamp score-then-update history. Review has 1,256 such positives among 730,784 queries (0.172%). We join these labels by query ID to the saved reciprocal ranks for all five primary seeds and logistic regression, with complete coverage and no duplicate IDs. The corrected diagnostic partitions reproduce the complete current means using the fixed query order and batch size. Group differences are descriptive, without a new bootstrap for each subgroup.

Table 16: Review validation grouped by strictly prior pair history. TG-Any averages five seeds. Gain share is each group's contribution to the overall TG-Any-minus-logistic-regression MRR difference, weighted by its query count.

Positive pair	Queries	TG-Any	Logistic reg.	Gain share
Previously unseen	729,528	0.1360	0.0363	99.43%
Repeated	1,256	0.4919	0.1615	0.57%

Almost all of the net gain comes from previously unseen pairs (Table 16). The global learned mixture also improves this group, reaching 0.1612 MRR; previously unseen pairs account for 99.42% of its net gain over logistic regression. These percentages allocate the count-weighted net MRR gain; they are not the fraction of queries that improved, nor a causal attribution to a channel.

E.10 Candidate-count sensitivity

Table 17 compares TG-Any and the three applicable same-feature controls at $Q \in \{20, 100, \text{full}\}$ on source `dev_gate`.

Table 17: Candidate-count sensitivity, `dev_gate`. "Full" is the official `dev_gate` negative count Q ($\approx$999); all values are current corrected scores, distinct from the historical K2 table. TG-Any values are 5-seed means.

Method	Q=20	Q=100	Full
LightGBM	0.8931	0.8611	0.5446
Logistic regression	0.8763	0.8554	0.8302
TG-Any (strict_zs)	0.8765	0.8576	0.8421
Best single channel	0.8509	0.8421	0.8331

E.11 Compute and efficiency

The base TG-Any conditioner has 18,584 parameters; `+logQ` has 18,712 and the linear ranker has 24. The corrected execution inventory contains 29 learned fits and ten control fit or selection jobs. These are execution counts, not a promise about final supplement file counts. Core learned jobs total 3,063.3 seconds of supervisor time; sensitivity learned jobs total 1,944.6 seconds under their supervisor. Source-control internal timers total 77.0 seconds, 250k target-control timers 445.3 seconds, and the Coin 1M LR timer 235.4 seconds. These five scopes differ and must not be summed into an end-to-end campaign runtime. The separate full scoring/statistics sequence took 24,242.6 seconds for its 418 sequential jobs, excluding previous feature construction, fits, and independent review. Shared-host execution affects wall time; these records are not a matched efficiency comparison with another architecture.

All prescribed corrected fits completed. LR emitted 1,458 RuntimeWarnings on the source, 1,878 on Coin 250k, 870 on Flight 250k, and 1,758 on Coin 1M (divide-by-zero, invalid, or overflow matrix multiplication). No ConvergenceWarning was recorded. The warnings and finite-output checks are preserved; completion alone does not establish that every numerical warning is harmless. Exact fit selection, environment, per-job timing, and failure records accompany the reproducibility evidence.

F Transfer to additional graphs

Coin and Flight extend the original Wiki, Review, and GoogleLocal experiments to recurrence-heavy, non-bipartite graphs. This appendix separates the historical expansion decisions from the corrected post-hoc estimates. The source, architectures, target coverage, and prescribed seeds are unchanged. All current comparisons use the corrected features, source maps, and refitted models and controls.

F.1 Original expansion design and preserved decisions

The expansion amendment `EXPANSION_PREREGISTRATION_AMENDMENT_20260727.md` and its hash-bound record `data/locks/expansion_preregistration_v1.json` fixed a validation criterion before the original evaluations. TG-Any's five-seed mean had to exceed the better of target-window-trained logistic regression and LightGBM. A passing target became eligible for a separately authorized official evaluation; the decision used the mean inequality, not a confidence interval. Coin passed and Flight failed. Those decisions and their independent review records remain unchanged: `EXPANSION_COIN_GATE_AUDIT_V1.md` and `EXPANSION_FLIGHT_GATE_AUDIT_V1.md`. Corrected values below neither rerun this gate nor authorize Flight's untested official split.

TG-Any and its source normalization maps are fitted on Wiki alone. The primary LR and LightGBM controls fit the last 250,000 target-training events, extended to complete timestamp buckets: 250,017 on Coin and 301,224 on Flight. These windows cover 1.5659% of Coin's 15,966,659 training events and 0.6403% of Flight's 47,043,612. They limit fitted query examples, not causal history: feature construction and negative sampling retain the earlier training prefix. The verified cutoff preserves the frozen sampler's results for retained queries. Full-stream fitting was outside the fixed budget; controls use fixed hyperparameters rather than target-validation model selection.

BestSingle is different: it selects one raw channel using the first 50,000 target-validation queries and fits no estimator. Coin selects channel 1 (zero-based index 0), Flight channel 5 (index 4). Full validation therefore overlaps channel selection. The separately reported strict suffix removes the entire boundary timestamp and all earlier queries without reselecting the channel (Appendix G.2). It is a post-selection sensitivity, not a new test.

Both targets use all supplied official validation candidates with $Q = 20$ negatives. This benchmark width was not reduced for compute; no target $Q \in \{20, 100, \text{full}\}$ sweep or additional sampler study was run. Comparisons match candidates within a target, but do not match distributions across graphs.

F.2 Corrected validation results

Table 18: Corrected post-hoc validation comparison with the stronger 250k-window fitted control (LR on each target). TG-Any reports a five-seed mean $\pm$ sample standard deviation. Differences and pointwise 95% temporal-block intervals condition on these fitted models. These estimates do not replace historical gate decisions.

Target	TG-Any	Target LR	Difference [95% CI]
Coin	0.7851 ± 0.0008	0.7837	+0.0013 [+0.0008, +0.0018]
Flight	0.8895 ± 0.0007	0.8941	-0.0046 [-0.0050, -0.0042]

Table 19: Expansion protocol. Training windows include complete timestamp buckets; every validation query retains the benchmark's 20 negatives and its positive.

Target	Validation queries	Control train rows	Q	Bipartite?
`tgbl-coin-v2`	3,421,416	250,017	20	No
`tgbl-flight-v2`	10,099,015	301,224	20	No

The current official TGB dataset documentation reports surprise indices 0.120 and 0.024 for Coin and Flight (Huang et al., 2023). We use these three-decimal values only as structural descriptors. Both v2 datasets corrected chronological-order errors; these rounded descriptors do not imply identical v1/v2 event streams or unrounded statistics.

F.3 Results by target

Coin: useful transfer, with a small official-set difference. On validation, TG-Any reaches 0.7851 $\pm$ 0.0008, exceeding 250k-window LR (0.7837) by +0.0013 with base interval [+0.0008, +0.0018]. All five seeds exceed this LR control. Global nevertheless reaches 0.7906, above the primary's 0.7851. The larger-window Coin LR reaches 0.784575; TG-Any's +0.000476 difference has base interval $[-0.000163, 0.001072]$. The study does not establish an advantage over unrestricted target training.

Historically, Coin's pass led to authorized official run `20260727T_expansion_coin_confirmatory` on 2026-08-06. Its receipt `results/expansion_eval/20260727T_expansion_coin_confirmatory/OFFICIAL_RECEIPT.json` is preserved as the original record, not evidence for the corrected estimates. The original exclusive-create and authorization records remain in `data/locks/EXPANSION_UNSEAL_COIN.json` and `data/locks/EXPANSION_CONFIRMATORY_COIN_CLAIMED.json`.

On the same 3,421,411 official queries, the corrected TG-Any mean is 0.8057 $\pm$ 0.0014, compared with 0.8056 for 250k-window LR. The difference is +0.0001 with interval [-0.0004, +0.0006], which includes zero. Per-seed MRRs are 0.8070, 0.8043, 0.8050, 0.8076, and 0.8048; only seeds 0 and 3 exceed LR (2/5). LightGBM scores 0.7038 and BestSingle 0.7823. These are post-hoc corrected scores on an already inspected official split, not renewed confirmation.

Flight: a validation-only transfer comparison. The complete corrected validation comparison evaluates 10,099,015 queries and all five primary seeds. TG-Any reaches 0.8895 $\pm$ 0.0007, compared with 0.8941 for 250k-window LR, 0.8927 for LightGBM, and 0.8773 for BestSingle. The primary-minus-LR difference is -0.0046 (base interval [-0.0050, -0.0042]). TG-Any is below both LR and LightGBM, with the paired intervals retaining that direction at all three prescribed block lengths, and above BestSingle. The historical failed mean gate remains the reason no Flight official test is included; no corrective outcome changes that coverage decision. The full fixed suffix and block-length sensitivity are reported separately.

For source context, Wiki gate MRR at $Q = 20$ is 0.8765 and at full width is 0.8421. These are comparisons of different graph and candidate distributions, not a causal test of recurrence or candidate width across targets.

F.4 What the additional graphs show

The five datasets comprise one training source and four transfer targets. Review and GoogleLocal show gains over the original source-trained same-feature controls. Coin and Flight compare a frozen source predictor with limited target-training windows; Coin's official-set LR difference is uncertain, while Global is stronger on target validation. Differences in graph structure, candidate sets, and fitting regimes prevent a causal attribution to any single graph property. Source-constant direct-overlap channels supply no candidate discrimination under the frozen Wiki maps (Appendix H).

G Supplementary comparisons and corrected analyses

These analyses were specified after the original target results and validation ordering were known. They are therefore post-hoc supplementary evidence, not a new test-entry gate or an independent confirmation. For the same-feature comparisons, the corrected campaign rebuilt the affected feature channel, refit the prescribed learned methods and controls, and rescored all compared methods under one fixed numerical runtime and candidate-batching policy. These comparisons do not use a legacy primary RR vector. The architecture comparison in Appendix G.3 pairs corrected TG-Any reciprocal ranks with the separately frozen CrossDyG outputs; CrossDyG retains its original execution policy and is not rerun. Historical outputs remain archived as historical evidence, including outcomes that disagree with the corrected comparisons.

G.1 Simple fusion rules on official targets

We compare the five corrected primary checkpoints with Uniform, all five corrected Global checkpoints, and the refitted source controls on the complete supplied candidate lists. Uniform has no fitted parameters; Global retains its source-only checkpoint selection and quantile map. Review official has 728,919 queries and GoogleLocal official has 43,546. The evaluation preserves padded masks, duplicate negative positions, and the exact average-tie rule.

Table 20: Post-hoc official-test comparisons with simple fusion rules. Every prespecified contrast is retained. Intervals are pointwise and conditional on the fitted seeds, using 10,000 paired resamples of complete timestamp blocks: 92 on Review and 44 on GoogleLocal at the base length. The supplement retains all half/base/double-length intervals and seed-level means.

Target	Contrast	ΔMRR	95% CI
Review	Uniform − primary	+0.1519	[+0.1498, +0.1540]
Review	Global − primary	+0.0316	[+0.0308, +0.0325]
GoogleLocal	Uniform − primary	-0.0245	[-0.0283, -0.0208]
GoogleLocal	Global − primary	-0.0010	[-0.0027, +0.0006]

On Review, the corrected primary mean is 0.161431, while Global and Uniform reach 0.193080 and 0.313302. On GoogleLocal, primary reaches 0.237860, Uniform reaches 0.213353, and Global reaches 0.236856; the pointwise interval for primary minus Global includes zero. Thus performance above the refitted same-feature controls does not imply an advantage over every simpler fusion rule. These are corrected, post-hoc comparisons on already inspected official cohorts and do not revise the historical test-entry decisions.

G.2 A fixed heuristic rule on target validation

We also transfer the LocalGlobal rule (Cornell et al., 2025), using its fixed descending lexicographic order of local recency, global recency, local popularity, and global popularity. This control has no fitted mixture weights and does not use the quantile bank. It shares the complete supplied candidate lists and strictly earlier target history with TG-Any. Every query in one timestamp group is scored before that group updates the four trackers, and duplicate negative positions are preserved. The comparison uses canonical candidate-list RR even for a previously unseen destination, rather than the historical source-anchor convention that assigned such a positive RR zero.

Table 21: Corrected five-seed primary versus canonical LocalGlobal on fixed validation cohorts. Δ is primary minus LocalGlobal. Intervals are base-scale pointwise temporal-block intervals conditional on the fitted primary seeds.

Cohort	Queries	Primary	LocalGlobal	ΔMRR	95% interval
Wiki gate	11,811	0.842086	0.835699	+0.006387	[0.004369, 0.008446]
Review val.	730,784	0.136599	0.139018	-0.002419	[-0.002578, -0.002260]
Coin val.	3,421,416	0.785050	0.780448	+0.004603	[0.003972, 0.005236]

The canonical Wiki rule is produced on all 23,621 validation queries and then aligned by exact ID to the 11,811-query gate; the equally sized Wiki official cohort is different. Relative to the historical source-anchor convention, canonical RR changes exactly 32 full-validation rows and 13 gate rows. The three contrasts are post-hoc; their intervals are pointwise and do not supply calibrated null-test p-values or a new validation gate. In particular, TG-Any is below LocalGlobal on Review even though it remains above the refitted same-feature source controls.

Post-selection Coin suffix. The full Coin validation result overlaps the first-shard target selection used by BestSingle. A separate sensitivity excludes the entire timestamp bucket at the boundary 1662128364: 38 rows span that boundary, leaving a 50,013-query prefix and a strict 3,371,403-query suffix. BestSingle remains the originally selected channel 0 (pair recency); no channel or model is reselected.

Table 22: Post-hoc Coin strict-suffix contrasts. Δ is corrected primary minus the named comparator.

Comparator	ΔMRR	Base 95% interval
Global	-0.005564	[-0.005914, -0.005220]
BestSingle (50k selection)	+0.033327	[0.031922, 0.034838]
Target logistic regression (250k)	+0.001373	[0.000829, 0.001878]
Target logistic regression (1M)	+0.000519	[-0.000131, 0.001129]

For the 1M logistic-regression comparison, the base and double-length intervals include zero while the half-length interval is positive. We retain that sensitivity rather than selecting the favorable scale. The suffix was defined after inspection and remains target-selected validation evidence; it is not a new test, an untouched gate, or a basis for retuning.

Post-selection Flight suffix. On all 10,099,015 Flight validation queries, the corrected primary reaches 0.889548, below target-window logistic regression at 0.894137 and LightGBM at 0.892723. The base primary-minus-control intervals are $[-0.005004, -0.004160]$ and $[-0.003995, -0.002344]$, respectively, and both directions persist at half and double block lengths.

The Flight sensitivity excludes the complete timestamp bucket at 1638162000. That 58,441-query bucket contains the first 50,000 selection queries and 8,441 later queries, leaving a strict suffix of 10,040,574. BestSingle remains the originally selected channel 4 (pair count); no channel or model is reselected.

Table 23: Post-hoc Flight strict-suffix contrasts. Δ is corrected primary minus the named comparator.

Comparator	ΔMRR	Base 95% interval
Global	-0.003053	[-0.003264, -0.002849]
BestSingle (50k selection)	+0.012217	[0.011714, 0.012717]
Target LightGBM (250k)	-0.003226	[-0.004049, -0.002416]
Target logistic regression (250k)	-0.004594	[-0.005014, -0.004183]

All four suffix directions persist at the half, base, and double block lengths. In particular, removing the selection prefix does not reverse the primary losses to logistic regression or LightGBM. Like the Coin analysis, this is a post-hoc sensitivity on target-selected validation data, not a new test or a new gate.

G.3 A fixed three-seed CrossDyG architecture comparison

The separately frozen adaptation of the CrossDyG author implementation (Pan et al., 2025b) completed all three source fits and all 730,784 Review validation queries per seed within its original resource budget. It uses raw causal target history, the supplied official candidate lists, and all-zero author-model node and edge features. It does not consume the TG-Any feature bank or normalization map. This is the disclosed source-only adaptation of the author implementation, not a reproduction claim for its published cross-graph results or a comparison using identical input features. No target outcome selected a checkpoint, seed, or resource decision.

We compare the fixed seeds 0–2 from each method on identical full query IDs, real timestamps, and candidate counts. TG-Any uses its corrected checkpoints and reciprocal ranks. Its three-seed mean here differs from the five-seed primary mean in Table 21; neither mean substitutes for the other. Both implementations use the pinned P1 package versions. CrossDyG retains its separately reviewed four-thread, per-query execution, while TG-Any uses one thread and batches of 64 complete queries.

Averaging the prescribed seeds within each query gives a primary-minus-CrossDyG MRR difference of +0.091403. The pointwise 95% temporal-block intervals are [0.089560, 0.093342], [0.089055, 0.093781], and [0.088344, 0.094474] at half, base, and double block lengths, respectively (138, 85, and 46 complete timestamp blocks). These intervals use 10,000 draws and condition on the fitted seeds; they do not measure uncertainty over new source or target graphs. The contrast is post-hoc

Table 24: Complete Review validation MRR for the fixed architecture comparison. Sample SD describes the three fitted seeds; queries are not replicated by seed.

Seed / summary	Corrected TG-Any	Adapted CrossDyG
0	0.135791	0.022580
1	0.137032	0.059935
2	0.136834	0.052933
Mean	0.136552	0.045149
Sample SD	0.000667	0.019857

validation evidence, not a new untouched test or a calibrated family rejection. It does not remove the primary model's losses to Uniform, Global, and LocalGlobal on this target.

The immutable ingestion seal binds the completed CrossDyG outputs after generation. The original scorer did not record generation-time reciprocal-rank hashes. Its saved ranks contain neither timestamps nor per-query negative or candidate counts. Timestamps and negative counts are joined by exact full query ID to the unchanged canonical validation stream; each seed's complete accounting separately confirms 730,784 queries and 73,809,184 candidates. The corrected comparison computes statistics from these saved outputs without new inference or training.

G.4 Complete-timestamp uncertainty analysis

The historical Review conditioning analysis and the historical Coin and Flight validation callers formed temporal blocks from global row indices rather than query timestamps. Those archived callers could split a timestamp group. This was an uncertainty-estimation defect in those historical artifacts; it is separate from the later preprocessing and numerical-runtime corrections that can also change scores and mean MRR.

The corrected campaign aligns every saved corrected RR column with canonical query IDs and timestamps, verifies complete coverage, and constructs blocks that end only after the complete timestamp group containing the requested length. Each resample draws whole blocks and divides the summed paired RR difference by the summed query count, so unequal blocks are weighted by their queries. The current contrasts are rebuilt from corrected RR columns; no legacy primary column is substituted into a corrected comparison.

Table 25: Complete-timestamp block sensitivity on validation. Effects condition on the five fitted primary seeds (and five Global seeds on Review). LR uses the fixed target-training window. Each accepted row uses 10,000 resamples; intervals are pointwise. The factor multiplies $L_0 = \max(1000, \lceil n/100 \rceil)$ before extending to a complete timestamp group. Flight's 250,000-row request expands to 301,224 rows because the frozen control window retains complete timestamp buckets.

Comparison	Factor	Blocks	ΔMRR	95% CI
Wiki: primary − LocalGlobal	0.5	24	+0.0064	[+0.0045, +0.0082]
	1.0	12	+0.0064	[+0.0044, +0.0084]
	2.0	6	+0.0064	[+0.0039, +0.0085]
Review: Global − primary	0.5	138	+0.0253	[+0.0247, +0.0259]
	1.0	85	+0.0253	[+0.0246, +0.0260]
	2.0	46	+0.0253	[+0.0244, +0.0262]
Coin: primary − LR (250k)	0.5	200	+0.0013	[+0.0009, +0.0017]
	1.0	100	+0.0013	[+0.0008, +0.0018]
	2.0	50	+0.0013	[+0.0007, +0.0019]
Flight: primary − LR	0.5	155	-0.0046	[-0.0049, -0.0043]
	1.0	82	-0.0046	[-0.0050, -0.0042]
	2.0	44	-0.0046	[-0.0051, -0.0041]

The half-, base-, and double-length analyses are all retained. Their intervals are pointwise and conditional on the fixed fitted seeds, rather than simultaneous confidence statements or unconditional training-seed inference. Bootstrap tail fractions are descriptive properties of the saved resamples, not calibrated null-test p-values. Changing the block length does not create independent graphs, and none of these post-hoc analyses opens a new gate. This manuscript integration reads the accepted small reports; it does not redraw the bootstrap samples.

G.5 Matched normalization transfer

We compare four complete source-fitted recipes on the Wiki source gate and the full Review and Coin validation cohorts. Quantile, z-score, log-only, and raw values each use the fixed seeds $\{0, 1, 2\}$ and their own source-only transform receipt and fitted checkpoint. We do not exchange a map under a model trained with another map, fit target statistics, apply additional clipping beyond the specified normalization recipes, or choose seeds from target outcomes. All twelve corrected models complete with finite scores.

Table 26: Matched normalization recipes, all using seeds $\{0, 1, 2\}$. Wiki is the 11,811-query source gate; Review and Coin are full target validation (730,784 and 3,421,416 queries). Each frozen map is paired with its own source-trained checkpoints. Mean $\pm$ sample seed standard deviation is reported for every fixed recipe.

Source-fitted recipe	Wiki gate	Review val.	Coin val.
Quantile	0.8422 ± 0.0003	0.1366 ± 0.0007	0.7849 ± 0.0005
Z-score	0.8321 ± 0.0017	0.1180 ± 0.0026	0.3140 ± 0.0683
Log-only	0.8068 ± 0.0021	0.1522 ± 0.0006	0.5516 ± 0.0709
Raw values	0.8341 ± 0.0018	0.0680 ± 0.0359	0.4683 ± 0.0433

Table 27: All six normalization-transfer contrasts at the base block length. Pointwise 95% intervals condition on the three fitted seeds per recipe and use 10,000 whole-timestamp block resamples (85 blocks on Review, 100 on Coin). All interval signs persist at half and double block lengths; complete intervals and seed results accompany the code.

Target	Contrast	ΔMRR	95% CI
Review	Z-score − Quantile	-0.0185	[-0.0197, -0.0174]
Review	Log-only − Quantile	+0.0156	[+0.0152, +0.0161]
Review	Raw values − Quantile	-0.0686	[-0.0705, -0.0667]
Coin	Z-score − Quantile	-0.4709	[-0.4747, -0.4674]
Coin	Log-only − Quantile	-0.2334	[-0.2361, -0.2309]
Coin	Raw values − Quantile	-0.3166	[-0.3202, -0.3131]

The corrected source ordering is quantile, raw, z-score, then log-only. In particular, the raw source-gate seed values are 0.836244, 0.833088, and 0.833085; the seed-0 collapse described for an archived historical run is not a corrected-campaign result. On Review, log-only exceeds matched quantile by 0.015636 MRR, with a base pointwise interval $[0.015202, 0.016065]$. On Coin, quantile remains strongest, and z-score, log-only, and raw trail it by 0.470947, 0.233375, and 0.316649 MRR, respectively. All three prescribed block factors retain these directions.

These are comparisons of trained recipes, so they combine effects on optimization and on the transferred representation. They do not isolate the causal effect of channel 24, normalization sampling, or numerical execution while holding fitted parameters fixed. Quantile remains the fixed primary recipe; no target result selects a replacement or creates a new gate.

G.6 Fixed feature diagnostics

These post-hoc diagnostics cover the complete Wiki source gate (`tgbl-wiki-v2`, `dev_gate`) and the full Review and Coin validation cohorts. They use the corrected feature bank and the fixed source-fitted maps. They fit no target statistic, select no seed, and do not rescore a model. The source gate

is distinct from the source TRAIN split used to fit the maps. The measurements are descriptive and do not assign a causal performance effect to a channel or a distribution shift.

Table 28: Complete recurrence partition. Each query appears in exactly one of the four middle columns. *Seen* means that the pair occurred in strictly earlier history. The last column instead divides recurrent negative candidate positions by all valid negative positions, retaining duplicates.

Split	Queries	Unseen + no seen −	Unseen + seen −	Seen + no seen −	Seen + seen −	Recurrent − (%)
Wiki gate	11,811	441	716	3,866	6,788	0.760069
Review val.	730,784	112,003	617,525	96	1,160	8.687456
Coin val.	3,421,416	101,051	371,202	516,651	2,432,512	35.398202

Recurrence is the raw repeat-indicator channel 7, evaluated with complete timestamp groups. Positive-pair recurrence and negative-candidate recurrence have different denominators. In Review, 1,256 of 730,784 positive pairs are recurrent (0.171870%); the unused historical count of 10,280 came from a different definition and is not combined with this partition. On that split, unseen positives account for 99.432429% of the primary-minus-source-LR net gain and seen positives for 0.567571%. For Global minus source LR the shares are 99.415954% and 0.584046%. These are decompositions of net RR differences, not percentages of individually improved queries or causal attributions. They do not overturn the primary losses to Global, Uniform, and LocalGlobal on Review.

Table 29: Raw direct-neighborhood channels: mean within-query candidate variance, with the percentage of nonconstant queries in parentheses. The fixed source quantile map makes all four mapped channels constant on all three cohorts.

Channel (one-based index)	Wiki gate	Review val.	Coin val.
Direct CN (12)	0 (0%)	0.0063208676 (30.5146%)	0.5836628743 (74.7546%)
Direct Jaccard (13)	0 (0%)	0.0000135656 (30.5146%)	0.0015457446 (74.7546%)
Direct Adamic–Adar (14)	0 (0%)	0.0002724455 (30.5146%)	0.0090068136 (74.7546%)
Direct recency-CN (15)	0 (0%)	0.0002245544 (29.8816%)	0.4498647554 (74.7546%)

The variance uses the population denominator over each query's valid positive and negative candidates and then averages queries equally. All three cohorts also have raw candidate-constant source recency (3), source rhythm alignment (4), source activity count (11), source novelty rate (23), and source local timescale (24). Wiki additionally has raw-constant direct channels 12–15. After source quantile mapping, the same nine channels are candidate-constant on every cohort. Raw target variation in Table 29 therefore does not establish mapped ranking variation.

Channel 24 is a corrected context scalar shared by every valid candidate in a query. Its raw and mapped within-query variances and positive-minus-negative gaps are exactly zero on all three cohorts. In exact arithmetic its direct additive term cannot distinguish candidates, although the conditioner can use the context to alter weights on other channels; the official scorer retains its exact floating-point tie rule. This observation neither isolates the effect of the correction nor implies that refitting was unnecessary.

Table 30: Largest absolute finite transformed value / nonfinite count over all valid values. Large z-scores are retained finite outcomes.

Split	Z-score	Log-only	Raw values
Wiki gate	39.5712 / 0	7.34407 / 0	1,546 / 0
Review val.	2.0000×10^{13} / 0	8.69433 / 0	5,968 / 0
Coin val.	2.0000×10^{13} / 0	13.48658 / 0	719,693 / 0

For each of the 24 channels, the conditioner summarizes the candidate mean, population standard deviation, maximum, effective candidate fraction, and the gap between the largest value and the largest strictly smaller value. The fixed histograms use 20 bins spanning $[-10^{-6}, 1 + 10^{-6}]$ plus

an explicit 21st residual category. Thus no finite out-of-range value is clipped or silently dropped. Table 31 reports the equal-channel mean total variation from Wiki.

Table 31: Mean total variation over 24 source-quantile summary histograms, including the residual category.

Query summary	Review vs. Wiki	Coin vs. Wiki
Mean	0.40284293	0.38376782
Standard deviation	0.18790141	0.27487389
Maximum	0.49992261	0.24715128
Effective candidate fraction	0.01487612	0.06419498
Distinct top gap	0.20918804	0.18322289

Only the effective candidate fraction has residual counts: five of 24 Wiki channels have a nonzero residual, with a maximum query fraction of 19.955973%; seven Review channels do, with a maximum of 6.142444%; Coin has none. The other four summary types have zero residual for every channel on every cohort. Total variation for the effective candidate fraction therefore includes a fixed-bin floating-point boundary effect as well as distributional differences and cannot by itself establish representation shift. No channel is selected by its apparent shift, and the five summary types are not averaged into a single diagnostic score.

G.7 Review's released node representation

Review is semantically a user–product review network. Its effective `py-tgb 2.2.0` loader, however, maps CSV source and destination tokens through one shared first-seen dictionary; the official negatives use the resulting numeric IDs. Our loader preserves this released representation. In the 4,144,621-row TRAIN-plus-validation prefix, 279,134 of 350,829 distinct tokens appear in both roles. This is a property of the benchmark encoding, not evidence that a real user and product are the same entity.

Consequently, the opposite-side disjointness argument in Appendix A does not apply to every released Review candidate pair. Validation query global row 3,413,838, for example, has source ID 330888 and official negative ID 274301 at candidate position 50. Their strictly earlier retained neighborhoods share ID 184027 through two training edges, and the bound raw cache has Direct CN equal to one at this position. The independently accepted corrected diagnostics find raw Direct CN candidate variation in 30.5146% of Review queries, while the fixed source quantile map makes channels 12–15 constant. The representation fact and witness therefore explain why nonzero raw direct features are possible without implying usable mapped variation, a causal feature contribution, or a reason to change the official candidate set.

H Channel mixture weights

We summarize the softmax weights of the corrected primary model on the complete Wiki `dev_gate` split (11,811 queries) and Review validation (730,784 queries). For each of the five fixed seeds, we divide the recorded channel-weight sums by the full query count, then compute the mean and sample standard deviation across those five query-averaged vectors. The weights were collected during the complete corrected scoring runs, without subsampling. Each channel's weight is shared across the candidates within a query.

Pair recency receives the largest mean weight on Wiki, while pair count dominates on Review. The mixture changes its allocation across graphs, but these averages do not establish a benefit from candidate conditioning: Global achieves higher MRR on complete Review validation. Seed variation is substantial for several channels, so the mean allocation is not a description of a typical individual model in every case.

All four direct channels are constant after the fitted source quantile normalization (Appendix A). Their contributions therefore add the same offset to every candidate's score. The source-recency and source-activity channels are also constant across candidates for a fixed query. A large weight

Table 32: Top five channels by corrected mean mixture weight on each full split. The standard deviation describes variation between the five fixed seed means; it is not a confidence interval. The table omits the other 19 channels.

Channel	Mean weight	Seed SD
Wiki `dev_gate`, $n = 11{,}811$		
`pair_recency`	0.483014	0.037844
`pair_count`	0.171822	0.021053
`direct_jaccard`	0.111431	0.152272
`source_recency`	0.053501	0.118230
`source_activity_count`	0.048495	0.107746
Review validation, $n = 730{,}784$		
`pair_count`	0.588174	0.042156
`pair_recency`	0.148907	0.020204
`direct_jaccard`	0.078038	0.106858
`cooccurrence_jaccard`	0.048965	0.006651
`source_recency`	0.035276	0.077318

on any such channel is not evidence of a ranking benefit. More generally, mixture weights describe allocation; identifying a channel's causal contribution would require an intervention.

For a fixed query, let C collect candidate-constant channels and let $a = \sum_{k \notin C} \alpha_k > 0$. Its score can be written as $s_i = b + a r_i$, where $b = \sum_{k \in C} \alpha_k \widetilde{x}_k$ and $r_i = \sum_{k \notin C} (\alpha_k / a) \widetilde{x}_{ik}$. In exact arithmetic, subtracting b and dividing by a preserve all ranks and ties. If $a = 0$, all candidates tie instead. Weight on constant channels therefore does not by itself prove rank degradation. Finite-precision evaluation can create or remove exact ties; the empirical comparisons use the prescribed common numerical runtime and the benchmark's exact tie rule. In the training loss, the same decomposition gives an effective temperature T_{loss}/a, whose optimization effect is not identified by these weight averages.

I Implementation and Reproducibility

This appendix distinguishes historical reproduction records from the corrected execution described in Appendix C. Run identifiers locate the original configurations and outputs; corrected records separately bind the revised features, fitted models and scores.

I.1 Supplement contents and validation

The independently checked code archive contains 29 learned checkpoints, seven serialized control models, three raw-channel BestSingle descriptors, and Uniform: 40 methods in the portable manifest. The archive also includes the corrected implementation and saved evidence for all 418 completed jobs, 375 scoring shards, ten aggregates, and the 31 downstream outputs specified by the fixed plan. Independent extraction verified the file inventory and content hashes. In the reference environment, prior checks of all 40 methods and four normalization maps matched the accepted runtime bitwise on small synthetic inputs. The final archive inherits these checks through byte-identical portable artifacts; no new numerical rerun was performed. Its README command was executed from the extracted root. These checks validate the delivered score-only interface, not a full rerun from raw graph data.

The validated portable entry point accepts already constructed, corrected 24-channel raw features and returns candidate scores in memory. Inputs are float32 arrays with complete candidate sets, one to 64 queries per batch, int32 negative counts, a fixed channel order, and NaN padding. It does not construct graph features, calculate reciprocal ranks, or issue a production scoring receipt. Synthetic checks cover finite values, masks, score types, transform bindings, and module provenance in the specified reference environment. Cross-platform or cross-BLAS bitwise equality is not established.

Raw events, full feature caches, and per-query reciprocal ranks remain external reproduction dependencies. The archive catalogs 841 static and 1,937 dynamic external files, including the separate Coin

1M training overlays and the complete CrossDyG comparison dependencies. Packaging checked their current paths and sizes and retained their recorded hash bindings; it did not reread and rehash all external arrays. Full evidence restoration requires those files and verification against the recorded identities. Newly computed delivery hashes are distinguished from inherited historical checksums. Anonymized sensitivity-model copies retain scientific parameters and disclose their path substitutions. Selected non-runtime JSON, Markdown, log, and Python evidence copies replace author-root and home-directory prefixes; runtime and accepted source-export bytes remain unchanged. The pinned CrossDyG checkout is obtained separately; its upstream source is not redistributed.

I.2 Corrected fitting and execution environment

Corrected execution uses CPython 3.10.20, NumPy 2.0.2, Torch 2.5.1, scikit-learn 1.5.2, SciPy 1.13.1, LightGBM 4.5.0 and joblib 1.5.3. The recorded resource-monitor version is psutil 7.2.2. The TG-Any and same-feature-control computations use one numerical thread. Training batches contain 256 queries; development and target scoring use batches of 64 complete queries. The original CrossDyG author implementation keeps its separately fixed four-thread, per-query execution policy.

The corrected fitting campaign completed 15 core executions (five each of the conditioned primary, Global, and $+\log Q$), five unconstrained linear executions, and nine alternative-normalization executions (three each of z-score, log-only and raw scores). Every execution uses all 110,178 source training queries and 11,810 source `dev_tune` queries. All 29 runs retain the first exact maximum of development MRR and stop after the configured five-epoch stale tail. The primary and Global parameterizations contain 18,584 parameters, $+\log Q$ contains 18,712, and the linear head contains 24. All prescribed seeds remain in the evidence, including the weakest completed alternative.

Ten control executions comprise source LR, LightGBM and raw-channel BestSingle; the corresponding three controls for each of Coin and Flight; and the additional Coin 1M LR. BestSingle selects a channel rather than fitting an estimator: source selection uses all 11,810 `dev_tune` queries, while each target selection uses its first 50,000 validation queries. The target selections overlap full validation, as disclosed in Appendix C. LR and LightGBM use fixed hyperparameters and the specified training windows; a generic receipt field named `selection_split` does not imply that these estimators were tuned on validation.

The LR receipts preserve NumPy matrix-multiplication RuntimeWarnings: 1,458 for source LR, 1,878 for Coin 250k, 870 for Flight 250k, and 1,758 for Coin 1M. No ConvergenceWarning was recorded, and the fits satisfy the frozen completion checks. Absence of a convergence warning alone does not establish the cause or harmlessness of the other warnings.

Per-job wall-time sums are 3,063.314 seconds for the 15 core executions, 1,944.629 for the 14 sensitivity executions, 76.974 for the three source controls, 445.301 for the six target-250k controls, and 235.446 for Coin 1M LR. These are five separate timer scopes, not a campaign total. Learned-run timers begin after resource waiting and include the isolated worker and supervisor verification. Source-control timers start before input preparation; target-control timers start before runtime and manifest checks. Feature construction, scoring, statistics and waiting are not all captured by these fit timers.

I.3 Base3 reference environment

The historical Base3 source-compatibility verification used Linux x86_64, CPython `3.10.20`, NumPy `1.23.5`, and single-threaded BLAS. The NumPy configuration and reference sort behavior matched the frozen environment checks. The synthetic suites passed all 12 and 8 cases, respectively.

I.4 Real-stream state trace

Before the historical validation scoring, `Base3SourceCompatV2` was compared with an independent oracle on wiki training and validation-positive events. The trace covered 119 batches, 23,621 validation events, and 25,263 candidates. All 101,052 compared state components matched, including EdgeBank membership, PopTrack arrays and ordering, t-CoMem deques and scores, and update-boundary states.

This comparison used training and validation-positive events, before validation scoring or access to its negative candidates.

I.5 Execution records

The P1-A reproduction run, state trace, and Base3 validation evaluation have content-hashed JSON receipts linking code, configuration, and outputs. Detailed execution logs are retained with those records.

I.6 Training recipe and source selection

The final configuration, `tgany_dev_v3.json`, trains five seeds (0–4) with 20 negatives per source-training query and reshuffles queries each epoch. Adam uses learning rate 3×10^{-4}, betas $(0.9, 0.999)$, epsilon 10^{-8}, and no weight decay. The InfoNCE temperature is 0.01. Batches contain 256 queries; training stops after at most 30 epochs or five epochs without a strict improvement in source `dev_tune` MRR, retaining the first checkpoint on an exact tie. The historical source-development choices below preceded the original `dev_gate` evaluation. The corrected campaign retains that recipe without repeating hyperparameter selection.

Removing the calibration network. The initial `tgany_dev_v1.json` enabled a shared per-channel calibration MLP, with T_{loss}=0.1 and learning rate 10^{-3}. Seed 0 reached a `dev_tune` MRR of 0.007233, close to the random-ranking expectation of approximately 0.00749 for 1,000 candidates. Training loss remained near chance, between 2.99 and 2.92.

In 300-step probes, disabling calibration reduced loss to 1.69, while the calibrated runs stayed at 3.01–3.04. A separate data-path check reproduced the best-single-channel `dev_tune` MRR of 0.845484. Version `tgany_dev_v2.json` therefore set `use_calibration=false`. The failed seed-0 record is retained at `results/tgany_dev/20260723T_p1c/seed0.json`.

Selecting temperature and learning rate. With calibration disabled and learning rate 3×10^{-4}, seed-0 `dev_tune` MRR was 0.7793 at T_{loss}=0.1, 0.8284 at T_{loss}=0.05, 0.8461 at T_{loss}=0.03, 0.8508 at T_{loss}=0.02, and 0.8521 at T_{loss}=0.01. We selected T_{loss}=0.01 for `v3`. A separate run with learning rate 10^{-3} and T_{loss}=0.2 reached 0.3088 and was discarded.

In the original campaign, the `v3` configuration was frozen before `dev_gate` evaluation. Historical five-seed training (P1-C) produced `dev_tune` MRRs of 0.85199–0.85233, with deterministic reproduction. The K2 source-development comparison followed under run `20260724T_k2`.

Correcting the controls' split label. Before K2, the controls runner used a noncanonical row-level `split` label where the loader expected `"validation"`. The mismatch caused rows to be dropped. We corrected the label, added a regression test, and regenerated every control under `20260724T_controls_final_v3` before using its value in K2.

I.7 GoogleLocal loader repair and receipt finalization

The original GoogleLocal campaign required a loader repair before scoring. That repaired loader was evaluated once in the original campaign; its receipt was subsequently finalized from saved outputs. This historical loader repair is distinct from the present feature and numerical-execution correction.

Source and destination ID handling. The original `20260724T_p2_confirmatory` loader assumed source IDs occupied the dense range $[0, \text{source_node_count})$. GoogleLocal's first source ID is 319,324, exceeding the registered count of 206,244 source nodes. Namespace validation therefore failed before any GoogleLocal feature or score was computed.

The repair was checked before results were accessed. First, a first-seen map assigned each raw source ID a dense index in chronological order. Second, review identified a mixed-indexing property of the pinned `tgb-seq` artifact: positive destination IDs include the upper bound destination_nodes, whereas negative IDs are zero-based with an exclusive upper bound. The positive destination check

was made inclusive, and the distinct-count check became an upper bound. Destination offset arithmetic and negative-candidate sampling and bounds were unchanged.

Independent verification reparsed all 1,913,967 CSV rows and confirmed the registered node count. It also checked row $56{,}892$, which contains the upper-bound destination and would have failed the old check. The code manifest was updated; wiki and review results were unchanged. The verification record is `refine-logs/P2_V2_GOOGLELOCAL_AUDIT_GO_20260724.md`.

Verified historical outputs. Run `20260724T_p2_confirmatory_v2_googlelocal` scored all 43,546 official queries. It evaluated three same-feature controls and five TG-Any seeds, completed the paired bootstrap, and passed the official `tgb-seq 0.1.2` check with maximum error 6.8×10^{-10}.

The final receipt was assembled from the saved per-query outputs after verifying their hashes and aggregate MRR. This produced `results/confirmatory/CONFIRMATORY_V2_GOOGLELOCAL_20260724T_p2_confirmatory_v2_googlelocal.json`. Receipt finalization did not reopen raw test data or rescore queries; the detailed record is `data/locks/p2_v2_r1_finalization_record.json`.

I.8 Official-test access

Wiki, review, and GoogleLocal test partitions remained isolated throughout development, apart from acquisition-time hashing and schema checks. Their metrics did not inform hyperparameters, model selection, or thresholds in the original campaign. Each original primary evaluation ran once under the protocols in §B.3 and §I.7, then resealed. Coin was evaluated once under a separate authorization on 2026-08-06 (Appendix F.3). The first additive Uniform/Global comparisons reused original primary reciprocal ranks (Appendix G); those mixed-runtime comparisons are retained as history. The present correction retrains the affected source predictors and refits controls, then rescores every method in the fixed matrix using the corrected inputs and common numerical policy. It does not reuse historical primary reciprocal ranks in a corrected matched comparison. These datasets have already been inspected, so the correction is post-hoc rather than a new untouched test. The original Coin/Flight test-entry decisions remain historical, and no Flight official test is added.