# ON THE POWER OF HEURISTICS IN TEMPORAL GRAPHS

**Filip Cornell**[*], **Oleg Smirnov**[*], **Gabriela Zarzar Gandler, Lele Cao**
King, part of Microsoft Gaming
{t-fcornell,oleg.smirnov,gabrielaz,lelecao}@microsoft.com

## ABSTRACT

Dynamic graph datasets often exhibit strong temporal patterns, such as recency, which prioritizes recent interactions, and popularity, which favors frequently occurring nodes. We demonstrate that simple heuristics leveraging only these patterns can perform on par or outperform state-of-the-art neural network models under standard evaluation protocols. To further explore these dynamics, we introduce metrics that quantify the impact of recency and popularity across datasets. Our experiments on BenchTemp (Huang et al., 2024a) and the Temporal Graph Benchmark (Huang et al., 2024b) show that our approaches achieve state-of-the-art performance across all datasets in the latter and secure top ranks on multiple datasets in the former. These results emphasize the importance of refined evaluation schemes to enable fair comparisons and promote the development of more robust temporal graph models. Additionally, they reveal that current deep learning methods often struggle to capture the key patterns underlying predictions in real-world temporal graphs. For reproducibility, we have made our code publicly available[1].

## 1 INTRODUCTION

Dynamic graphs model evolving real-world relationships, where nodes represent entities and edges capture their interactions. These graphs are dynamic, with nodes, edges, weights, or attributes continuously added, removed, or updated over time. Analyzing their temporal patterns is a critical challenge due to their broad applications in fields such as social networks and biological systems. To support this, challenging benchmarks using real-world datasets have been developed, facilitating efficient learning on dynamic graphs (Huang et al., 2024b;a). A key task in this domain is *link prediction*; to forecast future node links and is foundational for dynamic graph analysis.

Recent methods have increasingly focused on advanced neural network architectures for dynamic graph tasks (Kumar et al., 2019a; Xu et al., 2020; Rossi et al., 2020b; Wu et al., 2024; Gravina et al., 2024). However, dynamic graph datasets often exhibit strong recency and popularity patterns that can be effectively captured with simple memorization heuristics. Despite their simplicity, these heuristics have proven to be surprisingly robust baselines, frequently matching or outperforming more complex neural network-based approaches (Poursafaei et al., 2022b;a; Daniluk & Dąbrowski, 2023).

This work enhances the understanding of recency and popularity in temporal graphs by introducing heuristics that effectively capture multi-scale temporal patterns. These simple yet powerful methods demonstrate "unreasonable effectiveness", outperforming neural models in multiple datasets while also providing a scalable framework to analyze how temporal dynamics influence ranking behavior.

## 2 RELATED WORK

The effect of recency and popularity patterns has been extensively studied in the recommender system literature where it is typically attributed to selection, exposure, presentation, and other biases in interaction data (Chen et al., 2023; Wang et al., 2023; Klimashevskaia et al., 2024). Prior research on temporal patterns in dynamic graph datasets has focused on three main directions.

[*]Equal contribution.
[1]Available at https://openreview.net/forum?id=cTckUeh0Sw.

**Summary metrics.** A range of metrics has been developed to characterize the presence of various temporal patterns. For instance, Poursafaei et al. (2022a) characterized *novelty* (new edges per timestamp), *reoccurrence* (fraction of transductive edges), and *surprise* (test-only edges), demonstrating the challenge of predicting entirely new connections. Similarly, Daniluk & Dąbrowski (2023) proposed statistical distance-based measures to capture both short- and long-term global popularity dynamics, exposing weaknesses in existing evaluation protocols and negative sampling strategies.

**Tools for interpretation and visualization.** Complementing metrics are tools designed to make temporal patterns more interpretable. Poursafaei et al. (2022a) introduced Temporal Edge Appearance (TEA) and Temporal Edge Traffic (TET) plots, revealing when memorization-based approaches may fail – particularly in sparse graphs or when reoccurrence is low (i.e., a high surprise index). Shirzadkhani et al. (2024) later built on this work to provide deeper insights into data characteristics.

**Leveraging temporal heuristics for prediction.** Beyond measurement and visualization, researchers have proposed models and heuristics to exploit temporal information for prediction tasks. Poursafaei et al. (2022a) presented the EdgeBank heuristic, which achieves strong performance in transductive settings, while Daniluk & Dąbrowski (2023) introduced PopTrack, a simple popularity-based heuristic that outperformed state-of-the-art methods on multiple benchmarks which was then used to create harder negative samples. In a related vein, Poursafaei et al. (2022b) demonstrated that combining structural, interaction-based, and temporal features can produce expressive node representations for accurate classification in both static and dynamic scenarios.

## 3 METHOD

**The Notion of Recency.** Analyses of multiple benchmark datasets indicate that among various link prediction heuristics for dynamic graphs, *recency* (how recently a node has appeared as a destination) emerges as one of the most effective. In many real-world networks, recently active nodes often continue to participate, making recency a robust predictor. Moreover, frequent events also remain highly ranked through continually updated timestamps, reducing the need for added weighting. Building on those observations, the concept of recency is extended to multiple temporal scales, providing a more comprehensive perspective on dynamic graph behavior.

**Global Recency (GR).** This score identifies the most recently observed destination nodes across the entire graph. Instead of estimating a distribution, a simpler approach records each node's last appearance as a destination node, emphasizing temporal precision through memorization: $\text{GR}(v,t) = \max(\{-1\} \cup \{\tau \mid (u,v,\tau) \in \mathcal{G}, \tau < t\})$, where $\mathcal{G} \subset V \times V \times T$ is the set of temporal edges $(u,v,t)$, and $t \in T$ is a timestamp of the most recent destination occurrence of $v \in V$.

**Local Recency (LR).** This score captures the node-level temporal activity of individual destination nodes by focusing on their incoming connections. Rather than relying on a fixed time window, as in EdgeBank, each node retains a time-sorted list of its incoming nodes, effectively reflecting immediate temporal interactions: $\text{LR}(u,v,t) = \max(\{-1\} \cup \{\tau \mid (u,v,\tau) \in \mathcal{G}, \tau < t\})$, where $t$ is the timestamp of the most recent interaction between $u$ and $v$.

**The Notion of Popularity.** As highlighted by Daniluk & Dąbrowski (2023), many dynamic graph datasets exhibit a pronounced correlation with the historical popularity of destination nodes, reflecting a "rich-get-richer" dynamic in which frequently connected nodes continue to attract new links. This effect appears in various real-world systems, where once a node becomes popular, additional edges concentrate around it. Building on this insight, popularity-based heuristics can be implemented analogously to recency-based approaches, capturing how often nodes have served as popular destinations.

**Global Popularity (GP).** This score counts the total number of times $v$ has appeared as a destination node, being updated at each timestamp: $\text{GP}(v,t) = \sum_{\tau<t} \sum_{u' \in V} \mathbb{1}((u',v,\tau) \in \mathcal{G})$, where $\mathbb{1}(\cdot)$ is the indicator function that equals 1 if the condition holds, and 0 otherwise.

**Local Popularity (LP).** The score for a node $v$ with respect to a source node $u$ is defined as: $\text{LP}(u,v,t) = \sum_{\tau<t} \mathbb{1}((u,v,\tau) \in \mathcal{G})$, where the summation counts the number of times $v$ has appeared as a destination node specifically for source node $u$.

The pseudocode for the proposed heuristics is provided in Algorithm 1.

### 3.1 COMBINING HEURISTICS

Tailored heuristics are crucial for different datasets due to their unique characteristics. For example, the Local Recency heuristic performs poorly on the *tgbl-review* dataset because users rarely review the same product twice. This conflicts with the low *novelty* index (Poursafaei et al., 2022a) required for LR, as it cannot score unseen nodes for a given source. This highlights the need for complementary strategies. Insights from static graph methods, where heuristic combinations leverage individual strengths (Ma et al., 2024), suggest promising directions for dynamic graphs.

**Algorithm 1** Recency and Popularity Heuristics

**Require:** Temporal edges $(u, v, t) \in \mathcal{G}$, metric function $m(\cdot)$ (e.g., MRR)
Initialize LR, GR, LP, GP as empty dictionaries
**for** $t \in \mathcal{T}$ **do**
  **for** $(u, v) \in \mathcal{G}_t$ **do**
    **for** $h \in \{\texttt{LR}, \texttt{GR}, \texttt{LP}, \texttt{GP}\}$ **do**
      Compute $m(h, u, v, t)$
  **for** $(u, v) \in \mathcal{G}_t$ **do**
    $\texttt{LR}[u][v] \leftarrow t$
    $\texttt{GR}[v] \leftarrow t$
    $\texttt{LP}[u][v] \leftarrow \texttt{LP}[u][v] + 1$
    $\texttt{GP}[v] \leftarrow \texttt{GP}[v] + 1$
**return** Scores for LR, GR, LP, GP

The proposed algorithms also face challenges with ranking ties, which occur when multiple entities receive the same score. Unlike most machine learning models that produce continuous scores, $f\colon \mathcal{G} \to \mathbb{R}$, these heuristics rely on discrete scoring functions, such as counts or timestamps, i.e., $f\colon \mathcal{G} \to \mathbb{N}$. For recency-based heuristics, the frequency of ties is influenced by the granularity of dataset timing, with coarse-grained timestamps increasing the likelihood of identical scores. While ties are less common in sampled evaluations with fewer negative examples, they become more prevalent in full-ranking evaluations on datasets with coarser temporal resolution.

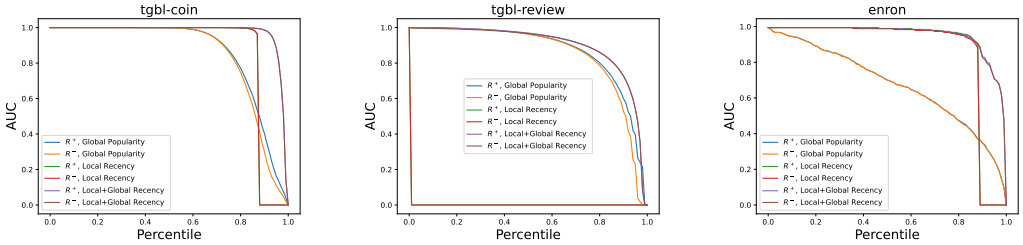
An approach in which heuristics are combined addresses this issue by stacking multiple heuristics into a product space, $f\colon \mathcal{G} \to \mathbb{N}^{|\mathcal{H}|}$, where $\mathcal{H} = \{h_1, h_2, \ldots, h_n\}$ is an ordered set of heuristics. When candidates share the same score under heuristic $h_i$, the next heuristic $h_{i+1}$ determines their internal ranks. This process iterates until ranks are fully resolved, or all heuristics are applied. Structuring the combination this way minimizes ranking ties, reduces discrepancies and improves prediction specificity across datasets. This approach applies to any heuristic in the family $\mathfrak{H}\colon \mathcal{G} \to S$, where $S \subseteq \mathbb{N}$. For recency heuristics, $S$ represents possible timestamps, while for popularity heuristics, $S = \{0, \ldots, |E|\}$, with $|E|$ as the number of edges. Selecting optimal heuristics for speed and performance depends on the dataset and is left for future study. In this work, unless stated otherwise, we use the order LR $\to$ GR $\to$ LP $\to$ GP.

## 4 EXPERIMENTS AND RESULTS

We evaluate the proposed approaches on the TGB (Huang et al., 2024b) and BenchTemp (Huang et al., 2024a) benchmarks using their respective proposed and standardized metrics (MRR with fixed set of negatives for TGB, AUC for BenchTemp). As shown in Table 1, the heuristic algorithms demonstrate competitive performance, achieving top positions on the TGB leaderboard[2] at the time of writing. Recency mostly outperforms popularity as a predictor across most datasets. However, popularity effectively resolves ties, serving as a complement to methods like LR. Detailed results in terms of both performance and runtime on BenchTemp and performance on TGB compared to baseline models are provided in Appendix A.

Two key observations emerge. First, heuristic approaches often outperform modern neural network methods when strong temporal patterns are present. Second, the same dataset, such as *Wikipedia*, can yield different metric values when evaluated under varying protocols, such as those used in TGB and BenchTemp. We hypothesize that these discrepancies stem from two main factors. First, neural network models may struggle to capture dominant temporal patterns, as they are often designed to prioritize long-term dependencies. Second, differences in evaluation protocols can highlight distinct aspects of the data, leading to inconsistencies, especially in sampled settings where results are highly sensitive to the number, quality, and selection of negative examples.

[2] https://tgb.complexdatalab.com/docs/leader_linkprop/




(a) TGB *tgbl-coin* dataset. (b) TGB *tgbl-review* dataset. (c) BenchTemp *Enron* dataset.

Figure 1: Complementary Normalized Rank (CNR) plots comparing optimistic ($R^+$) and pessimistic ($R^-$) ranks across various heuristics and their combinations. Each curve shows a method's performance using the CNR metric across percentiles of all edges, illustrating its ranking effectiveness.

| Dataset / Heuristic | LR | GR | LP | GP | Combined |
|---|---|---|---|---|---|
| **tgbl-coin** | 0.773 (1) | 0.613 (3) | 0.692 (3) | 0.726 (3) | 0.809 (1) |
| **tgbl-comment** | 0.164 (5) | 0.354 (4) | 0.106 (7) | 0.723 (1) | 0.455 (3) |
| **tgbl-flight** | 0.831 (1) | 0.603 (3) | 0.871 (1) | 0.183 | 0.88 (1) |
| **tgbl-review** | 0.001 | 0.321 | 0.001 | 0.394 (3) | 0.52 (1) |
| **tgbl-wiki** | 0.817 (1) | 0.04 (12) | 0.693 (5) | 0.157 (9) | 0.821 (1) |

Table 1: Mean Reciprocal Rank (MRR) on TGB test splits, with leaderboard rankings in parentheses.

## 5 Heuristics as Analysis Tools

The prevalence of recency and popularity patterns in a dataset is shaped by its underlying creation processes. To analyze their impact on ranking behavior, we introduce Complementary Normalized Rank (CNR) metric, computed using *optimistic* ($R^+$) and *pessimistic* ($R^-$) ranks (Ali et al., 2021; Huang et al., 2024b), where $R^+$ assumes favorable tie-breaking, and $R^-$ ranks tied candidates conservatively. At a given $p$, CNR is defined as $\text{CNR}(p) = 1 - R_p/|E|$, indicating that a fraction $p$ of edges was ranked at least as high as $R_p = |E|(1 - \text{CNR}(p))$. While not intended for direct method comparisons, this metric provides insights into dataset predictability and helps assess ranking effectiveness. As shown in Figure 1, CNR plots offer a comprehensive view of how temporal patterns shape ranking performance. While ranking all edges at every timestamp is infeasible for conventional methods, our heuristics enable efficient computation in logarithmic time with respect to $S$. Further implementation details are provided in Appendix B, and the discussion of CNR plots in Appendix C.

## 6 Conclusion

Our findings show that simple and highly efficient heuristics often outperform modern neural network approaches across a range of real-world benchmarks. Their effectiveness depends on dataset characteristics, while the methods introduced provide practical tools for interpreting these patterns and understanding temporal graph behavior. Inspired by recommender system research, we argue that accurately *modeling* recency and popularity patterns in temporal graph data may not always improve domain-specific metrics, as these patterns often reflect unintended biases such as selection, position, and exposure effects. We defer the exploration of de-biasing techniques for temporal graph datasets, better evaluation protocols, and methods for integrating heuristic signals into neural models to future work.

### Acknowledgement

This work was partially funded by Wallenberg AI, Autonomous Systems and Software Program (WASP).

## REFERENCES

Mehdi Ali, Max Berrendorf, Charles Tapley Hoyt, Laurent Vermue, Mikhail Galkin, Sahand Sharifzadeh, Asja Fischer, Volker Tresp, and Jens Lehmann. Bringing light into the dark: A large-scale evaluation of knowledge graph embedding models under a unified framework. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(12):8825–8845, 2021.

Jiawei Chen, Hande Dong, Xiang Wang, Fuli Feng, Meng Wang, and Xiangnan He. Bias and debias in recommender system: A survey and future directions. *ACM Transactions on Information Systems*, 41(3):1–39, 2023.

Weilin Cong, Si Zhang, Jian Kang, Baichuan Yuan, Hao Wu, Xin Zhou, Hanghang Tong, and Mehrdad Mahdavi. Do we really need complicated model architectures for temporal networks? In *The Eleventh International Conference on Learning Representations*, 2023.

Michał Daniluk and Jacek Dąbrowski. Temporal graph models fail to capture global temporal dynamics. *arXiv preprint arXiv:2309.15730*, 2023.

Peter M Fenwick. A new data structure for cumulative frequency tables. *Software: Practice and experience*, 24(3):327–336, 1994.

Alessio Gravina, Giulio Lovisotto, Claudio Gallicchio, Davide Bacciu, and Claas Grohnfeldt. Long range propagation on continuous-time dynamic graphs. In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24, 2024.

Qiang Huang, Xin Wang, Susie Xi Rao, Zhichao Han, Zitao Zhang, Yongjun He, Quanqing Xu, Yang Zhao, Zhigao Zheng, and Jiawei Jiang. Benchtemp: A general benchmark for evaluating temporal graph neural networks. In *2024 IEEE 40th International Conference on Data Engineering (ICDE)*, pp. 4044–4057. IEEE, 2024a.

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. *Advances in Neural Information Processing Systems*, 36, 2024b.

Ming Jin, Yuan-Fang Li, and Shirui Pan. Neural temporal walks: Motif-aware representation learning on continuous-time dynamic graphs. *Advances in Neural Information Processing Systems*, 35: 19874–19886, 2022.

Anastasiia Klimashevskaia, Dietmar Jannach, Mehdi Elahi, and Christoph Trattner. A survey on popularity bias in recommender systems. *User Modeling and User-Adapted Interaction*, 34(5): 1777–1834, 2024.

Walid Krichene and Steffen Rendle. On sampled metrics for item recommendation. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*, pp. 1748–1757, 2020.

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 and Data Mining*, KDD '19. ACM, July 2019a. doi: 10.1145/3292500.3330895.

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*, pp. 1269–1278, 2019b.

Yuhong Luo and Pan Li. Neighborhood-aware scalable temporal network representation learning. In *Learning on Graphs Conference*, pp. 1–1. PMLR, 2022.

Li Ma, Haoyu Han, Juanhui Li, Harry Shomer, Hui Liu, Xiaofeng Gao, and Jiliang Tang. Mixture of link predictors on graphs. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

Farimah Poursafaei, Shenyang Huang, Kellin Pelrine, and Reihaneh Rabbany. Towards better evaluation for dynamic link prediction. *Advances in Neural Information Processing Systems*, 35: 32928–32941, 2022a.

Farimah Poursafaei, Zeljko Zilic, and Reihaneh Rabbany. A strong node classification baseline for temporal graphs. In *Proceedings of the 2022 SIAM International Conference on Data Mining (SDM)*, pp. 648–656. SIAM, 2022b.

Emanuele Rossi, Ben Chamberlain, Fabrizio Frasca, Davide Eynard, Federico Monti, and Michael Bronstein. Temporal graph networks for deep learning on dynamic graphs. *arXiv preprint arXiv:2006.10637*, 2020a.

Emanuele Rossi, Ben Chamberlain, Fabrizio Frasca, Davide Eynard, Federico Monti, and Michael Bronstein. Temporal graph networks for deep learning on dynamic graphs. In *ICML 2020 Workshop on Graph Representation Learning*, 2020b.

Razieh Shirzadkhani, Shenyang Huang, Elahe Kooshafar, Reihaneh Rabbany, and Farimah Poursafaei. Temporal graph analysis with tgx. In *Proceedings of the 17th ACM International Conference on Web Search and Data Mining*, pp. 1086–1089, 2024.

Rakshit Trivedi, Mehrdad Farajtabar, Prasenjeet Biswal, and Hongyuan Zha. Dyrep: Learning representations over dynamic graphs. In *International conference on learning representations*, 2019.

Lu Wang, Xiaofu Chang, Shuang Li, Yunfei Chu, Hui Li, Wei Zhang, Xiaofeng He, Le Song, Jingren Zhou, and Hongxia Yang. Tcl: Transformer-based dynamic graph modelling via contrastive learning. *arXiv preprint arXiv:2105.07944*, 2021a.

Yanbang Wang, Yen-Yu Chang, Yunyu Liu, Jure Leskovec, and Pan Li. Inductive representation learning in temporal networks via causal anonymous walks. In *International Conference on Learning Representations (ICLR)*, 2021b.

Yifan Wang, Weizhi Ma, Min Zhang, Yiqun Liu, and Shaoping Ma. A survey on the fairness of recommender systems. *ACM Transactions on Information Systems*, 41(3):1–43, 2023.

Yuxia Wu, Yuan Fang, and Lizi Liao. On the feasibility of simple transformer for dynamic graph modeling. In *Proceedings of the ACM on Web Conference 2024*, pp. 870–880, 2024.

Da Xu, Chuanwei Ruan, Evren Korpeoglu, Sushant Kumar, and Kannan Achan. Inductive representation learning on temporal graphs. In *International Conference on Learning Representations*, 2020.

Le Yu, Leilei Sun, Bowen Du, and Weifeng Lv. Towards better dynamic graph learning: New architecture and unified library. *Advances in Neural Information Processing Systems*, 36:67686–67700, 2023.

Xiaohui Zhang, Yanbo Wang, Xiyuan Wang, and Muhan Zhang. Efficient neural common neighbor for temporal graph link prediction. *arXiv preprint arXiv:2406.07926*, 2024.

# APPENDIX

## A ADDITIONAL RESULTS

Table 2 presents TGB test MRR results alongside the official leaderboard[3], showing that our heuristics consistently outperform baseline methods across all cases. Table 3 reports AUC-ROC results for our heuristics compared to the BenchTemp leaderboard[4], highlighting substantial performance variability across datasets. These results reinforce that each heuristic's effectiveness depends on the presence of the specific temporal pattern it is designed to exploit.

[3] `https://tgb.complexdatalab.com/docs/leader_linkprop/`

[4] `https://my-website-6gnpiaym0891702b-1257259254.tcloudbaseapp.com/`

For TGB, we use baseline results from the official leaderboard, except for *tgbl-review* and *tgbl-subreddit*, as the former was updated and the latter was not included at the time of writing. In these cases, we conduct our own evaluations using the reference code, maintaining the TGB hyperparameter configurations. For BenchTemp, we adopt leaderboard baselines and ensure fair comparisons by applying its negative sampling method with a 3-fold evaluation for robust metric estimation.

In the corresponding benchmarks, we compare our approach against JODIE (Kumar et al., 2019b), NeurTW (Jin et al., 2022), DyGFormer (Yu et al., 2023), NAT (Luo & Li, 2022), TNCN (Zhang et al., 2024), CAWN (Wang et al., 2021b), TGN (Rossi et al., 2020a), TCL (Wang et al., 2021a), TGAT (Xu et al., 2020), DyRep (Trivedi et al., 2019), and GraphMixer (Cong et al., 2023).

| Dataset | | tgbl-wiki | tgbl-coin | tgbl-review | tgbl-comment | tgbl-flight | tgbl-subreddit |
|---|---|---|---|---|---|---|---|
| **Baseline Models** | DyGFormer | **0.798 ± 0.004** | 0.752 ± 0.004 | — | **0.670 ± 0.001** | — | — |
| | NAT | 0.749 ± 0.010 | — | — | — | — | — |
| | TNCN | 0.718 ± 0.001 | **0.762 ± 0.004** | — | **0.697 ± 0.006** | 0.820 ± 0.004 | — |
| | CAWN | 0.711 ± 0.006 | — | — | — | — | — |
| | EdgeBank$_{tw}$ | 0.571 | 0.580 | 0.020 | 0.149 | 0.387 | 0.589 |
| | EdgeBank$_{\infty}$ | 0.495 | 0.359 | 0.021 | 0.129 | 0.167 | 0.485 |
| | TGN | 0.396 ± 0.060 | 0.586 ± 0.037 | **0.414 ± 0.011** | 0.379 ± 0.021 | 0.705 ± 0.020 | 0.49 ± 0.022 |
| | TCL | 0.207 ± 0.025 | — | — | — | — | — |
| | TGAT | 0.141 ± 0.007 | — | 0.355 ± 0.012 | — | — | 0.388 ± 0.01 |
| | GraphMixer | 0.118 ± 0.002 | — | 0.255 ± 0.193 | — | — | 0.195 ± 0.001 |
| | DyRep | 0.050 ± 0.017 | 0.452 ± 0.046 | 0.106 ± 0.016 | 0.289 ± 0.033 | 0.556 ± 0.014 | 0.113 ± 0.022 |
| **Heuristics** | GR | 0.157 | 0.726 | **0.394** | **0.723** | 0.200 | 0.097 |
| | LR | **0.817** | **0.773** | 0.001 | 0.164 | **0.840** | **0.716** |
| | GP | 0.193 | 0.613 | 0.321 | 0.354 | 0.619 | 0.193 |
| | LP | 0.707 | 0.692 | 0.001 | 0.106 | **0.876** | **0.738** |
| | Combined | **0.821** | **0.809** | **0.522** | 0.455 | **0.88** | **0.717** |

Table 2: Comparison of the TGB (Huang et al., 2024b) leaderboard and the proposed heuristics using standardized test MRR, with the **best**, **second-best**, and **third-best** results highlighted in bold and color-coded.

## B Efficient Implementation of Heuristics

Computing top-$K$ ranking metrics, such as MRR, across an entire dataset is often computationally expensive. A brute-force approach to determine the exact rank of an entity requires scoring all entities against a query, resulting in a complexity of $\mathcal{O}(|V|)$. To speed up evaluation, many methods sample a smaller set of false (negative) examples and rank the positive item within this subset. However, such methods are biased and inconsistent estimators of true ranking metrics (Krichene & Rendle, 2020). Only AUC-ROC has been proven to provide consistent evaluations, where expected values converge to true performance as the sample size grows.

In contrast, the proposed heuristics enable efficient calculation of full rankings for arbitrary queries in $\mathcal{O}(\log S)$ time by leveraging optimized data structures. When scores are integer-based, each score effectively becomes an index in a consolidated list, grouping all nodes with the same score. This arrangement facilitates direct calculation of exact optimistic and pessimistic ranks by summing nodes in indices below (and, for optimistic ranks, equal to) a particular score. For recency-based heuristics, Fenwick Trees (Fenwick, 1994) are used for efficient ranking by storing and retrieving these contiguous sums, which reduces the worst-case complexity of computing full ranks from $\mathcal{O}(|V|)$ to $\mathcal{O}(\log S)$, while the memory usage is bounded by $\mathcal{O}(S + |V|)$ where $S$ represents the number of unique timestamps, and $|V|$ is the number of nodes.

In essence, the algorithm manages edge updates by dynamically tracking their occurrences across timestamps in both global and local settings. This design achieves a balance between precision and scalability, making it well-suited for large-scale temporal graph data. It should be noted that combining heuristics increases overall complexity, as their independence results in higher computational demands compared to using a single heuristic.

To demonstrate effectiveness, we present the runtime measurements in Table 4. For heuristic methods, we report the runtime for a single pass through the training combined with the average runtime through both test sets. Model runtimes are obtained from the BenchTemp leaderboard, where they are executed on GPUs, whereas heuristic methods are run on a CPU.

| Dataset | Heuristics | | | | | Baseline Models | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Combined | GR | LR | GP | LP | CAWN | DyRep | JODIE | NAT | NeurTW | TGAT | TGN |
| **Inductive** | | | | | | | | | | | | |
| **CanParl** | 0.632 ± 0.002 | 0.627 ± 0.002 | **0.655 ± 0.001** | 0.630 ± 0.003 | 0.653 ± 0.001 | **0.715 ± 0.097** | 0.553 ± 0.009 | 0.501 ± 0.016 | 0.621 ± 0.073 | **0.887 ± 0.014** | 0.580 ± 0.007 | 0.573 ± 0.027 |
| **CollegeMsg** | **0.934 ± 0.000** | 0.921 ± 0.001 | 0.838 ± 0.001 | 0.788 ± 0.002 | 0.836 ± 0.001 | 0.916 ± 0.004 | 0.484 ± 0.012 | 0.510 ± 0.031 | **0.960 ± 0.017** | **0.973 ± 0.000** | 0.715 ± 0.001 | 0.777 ± 0.052 |
| **Contact** | **0.978 ± 0.000** | 0.875 ± 0.000 | 0.966 ± 0.000 | 0.623 ± 0.001 | 0.939 ± 0.000 | 0.969 ± 0.003 | 0.865 ± 0.043 | 0.936 ± 0.003 | 0.947 ± 0.013 | **0.984 ± 0.000** | 0.557 ± 0.005 | **0.952 ± 0.006** |
| **Enron** | **0.940 ± 0.002** | 0.811 ± 0.003 | **0.918 ± 0.001** | 0.540 ± 0.004 | 0.910 ± 0.003 | **0.916 ± 0.002** | 0.712 ± 0.060 | 0.804 ± 0.022 | **0.952 ± 0.006** | 0.905 ± 0.003 | 0.553 ± 0.015 | 0.816 ± 0.023 |
| **Flights** | **0.927 ± 0.000** | 0.784 ± 0.001 | 0.851 ± 0.000 | 0.827 ± 0.001 | 0.850 ± 0.000 | **0.983 ± 0.000** | 0.869 ± 0.013 | 0.922 ± 0.009 | **0.983 ± 0.003** | 0.916 ± 0.000 | 0.832 ± 0.004 | **0.952 ± 0.004** |
| **LastFM** | **0.910 ± 0.000** | 0.760 ± 0.001 | 0.902 ± 0.000 | 0.491 ± 0.000 | 0.889 ± 0.000 | **0.908 ± 0.002** | 0.799 ± 0.044 | 0.801 ± 0.034 | **0.914 ± 0.004** | 0.884 ± 0.000 | 0.520 ± 0.014 | 0.828 ± 0.014 |
| **MOOC** | 0.707 ± 0.000 | 0.694 ± 0.001 | 0.664 ± 0.000 | 0.533 ± 0.001 | 0.616 ± 0.000 | **0.948 ± 0.001** | **0.827 ± 0.018** | 0.778 ± 0.058 | 0.733 ± 0.043 | 0.804 ± 0.022 | 0.737 ± 0.006 | **0.887 ± 0.025** |
| **Reddit** | **0.994 ± 0.008** | 0.904 ± 0.042 | 0.933 ± 0.000 | 0.799 ± 0.045 | 0.933 ± 0.000 | **0.987 ± 0.000** | 0.958 ± 0.000 | 0.951 ± 0.005 | **0.991 ± 0.003** | 0.980 ± 0.001 | 0.965 ± 0.000 | 0.976 ± 0.000 |
| **SocialEvo** | **0.952 ± 0.001** | 0.786 ± 0.000 | **0.952 ± 0.000** | 0.481 ± 0.002 | 0.861 ± 0.000 | **0.930 ± 0.000** | 0.916 ± 0.004 | 0.896 ± 0.023 | 0.896 ± 0.016 | – | 0.675 ± 0.002 | **0.924 ± 0.008** |
| **Taobao** | 0.219 ± 0.001 | 0.213 ± 0.001 | 0.521 ± 0.000 | 0.201 ± 0.002 | 0.521 ± 0.000 | 0.774 ± 0.003 | 0.703 ± 0.001 | 0.701 ± 0.001 | **0.999 ± 0.000** | **0.884 ± 0.002** | 0.526 ± 0.012 | **0.774 ± 0.003** |
| **UCI** | **0.932 ± 0.001** | 0.920 ± 0.001 | 0.835 ± 0.000 | 0.783 ± 0.004 | 0.833 ± 0.000 | 0.918 ± 0.002 | 0.430 ± 0.043 | 0.752 ± 0.006 | **0.962 ± 0.017** | **0.969 ± 0.003** | 0.702 ± 0.005 | 0.808 ± 0.024 |
| **UNTrade** | 0.632 ± 0.003 | 0.552 ± 0.001 | **0.691 ± 0.001** | 0.473 ± 0.003 | 0.594 ± 0.001 | **0.740 ± 0.001** | 0.647 ± 0.011 | 0.673 ± 0.013 | 0.648 ± 0.066 | 0.592 ± 0.033 | – | **0.673 ± 0.013** |
| **UNVote** | 0.54 ± 0.000 | 0.517 ± 0.000 | **0.639 ± 0.001** | 0.374 ± 0.001 | 0.499 ± 0.001 | **0.591 ± 0.001** | 0.499 ± 0.010 | 0.512 ± 0.001 | **0.779 ± 0.008** | 0.586 ± 0.000 | 0.477 ± 0.005 | 0.588 ± 0.012 |
| **USLegis** | 0.58 ± 0.002 | 0.604 ± 0.002 | 0.576 ± 0.004 | 0.375 ± 0.003 | 0.539 ± 0.004 | **0.967 ± 0.003** | 0.598 ± 0.010 | 0.584 ± 0.013 | **0.745 ± 0.029** | **0.971 ± 0.001** | 0.557 ± 0.008 | 0.613 ± 0.005 |
| **Wikipedia** | 0.963 ± 0.001 | 0.916 ± 0.000 | 0.917 ± 0.000 | 0.431 ± 0.001 | 0.917 ± 0.000 | **0.989 ± 0.000** | 0.910 ± 0.003 | 0.931 ± 0.002 | **0.996 ± 0.003** | **0.990 ± 0.000** | 0.934 ± 0.003 | 0.978 ± 0.001 |
| **Inductive (New–New)** | | | | | | | | | | | | |
| **CanParl** | 0.566 ± 0.004 | 0.564 ± 0.003 | **0.643 ± 0.004** | 0.561 ± 0.000 | 0.644 ± 0.000 | **0.701 ± 0.124** | 0.443 ± 0.007 | 0.435 ± 0.009 | 0.569 ± 0.033 | **0.888 ± 0.005** | 0.596 ± 0.007 | 0.563 ± 0.040 |
| **CollegeMsg** | **0.931 ± 0.001** | 0.920 ± 0.003 | 0.843 ± 0.001 | 0.729 ± 0.003 | 0.842 ± 0.000 | 0.930 ± 0.002 | 0.527 ± 0.005 | 0.532 ± 0.027 | **0.940 ± 0.037** | **0.976 ± 0.001** | 0.783 ± 0.003 | 0.797 ± 0.011 |
| **Contact** | **0.979 ± 0.001** | 0.872 ± 0.003 | **0.968 ± 0.001** | 0.460 ± 0.004 | 0.952 ± 0.002 | 0.965 ± 0.001 | 0.660 ± 0.040 | 0.753 ± 0.006 | 0.949 ± 0.003 | **0.982 ± 0.000** | 0.545 ± 0.006 | 0.912 ± 0.005 |
| **Enron** | **0.966 ± 0.001** | 0.773 ± 0.004 | **0.952 ± 0.004** | 0.404 ± 0.009 | 0.953 ± 0.001 | **0.961 ± 0.005** | 0.657 ± 0.052 | 0.680 ± 0.002 | **0.969 ± 0.005** | 0.939 ± 0.000 | 0.531 ± 0.018 | 0.764 ± 0.018 |
| **Flights** | 0.918 ± 0.001 | 0.771 ± 0.003 | 0.858 ± 0.000 | 0.717 ± 0.003 | 0.876 ± 0.000 | **0.987 ± 0.001** | 0.890 ± 0.027 | 0.930 ± 0.008 | **0.991 ± 0.001** | 0.941 ± 0.000 | 0.857 ± 0.006 | **0.965 ± 0.002** |
| **LastFM** | **0.966 ± 0.000** | 0.910 ± 0.000 | 0.962 ± 0.000 | 0.500 ± 0.001 | 0.962 ± 0.001 | **0.970 ± 0.000** | 0.868 ± 0.016 | 0.885 ± 0.009 | **0.974 ± 0.002** | 0.963 ± 0.000 | 0.509 ± 0.033 | 0.875 ± 0.001 |
| **MOOC** | 0.688 ± 0.001 | 0.669 ± 0.003 | 0.647 ± 0.003 | 0.314 ± 0.009 | 0.613 ± 0.000 | **0.942 ± 0.000** | 0.722 ± 0.018 | 0.707 ± 0.016 | 0.656 ± 0.029 | **0.805 ± 0.009** | 0.740 ± 0.006 | **0.876 ± 0.004** |
| **Reddit** | **1.000 ± 0.000** | 0.938 ± 0.088 | 0.875 ± 0.000 | 0.979 ± 0.030 | 0.875 ± 0.000 | **0.995 ± 0.002** | 0.953 ± 0.005 | 0.938 ± 0.009 | **0.995 ± 0.001** | **0.988 ± 0.000** | 0.960 ± 0.004 | 0.981 ± 0.000 |
| **SocialEvo** | **0.950 ± 0.001** | 0.784 ± 0.006 | **0.951 ± 0.001** | 0.237 ± 0.004 | 0.901 ± 0.001 | **0.932 ± 0.000** | 0.774 ± 0.022 | 0.648 ± 0.049 | **0.928 ± 0.047** | – | 0.466 ± 0.007 | 0.879 ± 0.004 |
| **Taobao** | 0.170 ± 0.002 | 0.164 ± 0.001 | 0.522 ± 0.000 | 0.133 ± 0.001 | 0.522 ± 0.000 | **0.785 ± 0.015** | 0.717 ± 0.001 | 0.717 ± 0.001 | **1.000 ± 0.000** | **0.908 ± 0.001** | 0.523 ± 0.005 | 0.708 ± 0.001 |
| **UCI** | **0.929 ± 0.001** | 0.917 ± 0.002 | 0.840 ± 0.001 | 0.724 ± 0.003 | 0.838 ± 0.000 | 0.924 ± 0.003 | 0.477 ± 0.010 | 0.639 ± 0.016 | **0.947 ± 0.026** | **0.972 ± 0.002** | 0.768 ± 0.004 | 0.805 ± 0.021 |
| **UNTrade** | 0.587 ± 0.018 | 0.554 ± 0.003 | **0.704 ± 0.006** | 0.262 ± 0.014 | 0.703 ± 0.010 | **0.746 ± 0.008** | 0.536 ± 0.015 | 0.592 ± 0.009 | **0.688 ± 0.018** | 0.594 ± 0.060 | – | 0.507 ± 0.006 |
| **UNVote** | 0.379 ± 0.001 | 0.516 ± 0.001 | **0.634 ± 0.002** | 0.145 ± 0.004 | **0.615 ± 0.003** | 0.578 ± 0.002 | 0.473 ± 0.003 | 0.491 ± 0.020 | **0.720 ± 0.075** | 0.567 ± 0.000 | 0.500 ± 0.006 | **0.634 ± 0.002** |
| **USLegis** | 0.442 ± 0.002 | 0.490 ± 0.003 | 0.538 ± 0.004 | 0.175 ± 0.003 | 0.538 ± 0.004 | **0.974 ± 0.006** | 0.564 ± 0.019 | 0.539 ± 0.008 | 0.890 ± 0.022 | **0.979 ± 0.000** | 0.532 ± 0.029 | **0.890 ± 0.022** |
| **Wikipedia** | 0.976 ± 0.001 | 0.929 ± 0.001 | 0.940 ± 0.000 | 0.352 ± 0.002 | 0.940 ± 0.000 | **0.993 ± 0.001** | 0.926 ± 0.003 | 0.935 ± 0.005 | **0.998 ± 0.001** | **0.996 ± 0.000** | 0.958 ± 0.004 | 0.986 ± 0.001 |
| **Inductive (New–Old)** | | | | | | | | | | | | |
| **CanParl** | **0.648 ± 0.002** | 0.641 ± 0.002 | 0.539 ± 0.020 | 0.645 ± 0.004 | 0.641 ± 0.002 | **0.723 ± 0.085** | 0.507 ± 0.001 | 0.508 ± 0.001 | **0.628 ± 0.081** | **0.885 ± 0.010** | 0.572 ± 0.006 | 0.569 ± 0.022 |
| **CollegeMsg** | **0.933 ± 0.000** | 0.921 ± 0.003 | 0.481 ± 0.028 | 0.807 ± 0.003 | 0.921 ± 0.003 | 0.917 ± 0.003 | 0.517 ± 0.036 | 0.832 ± 0.001 | **0.973 ± 0.019** | **0.968 ± 0.002** | 0.701 ± 0.005 | 0.772 ± 0.037 |
| **Contact** | **0.978 ± 0.000** | 0.876 ± 0.000 | 0.857 ± 0.045 | 0.632 ± 0.001 | 0.876 ± 0.000 | **0.969 ± 0.003** | 0.935 ± 0.003 | 0.934 ± 0.003 | 0.935 ± 0.020 | **0.984 ± 0.000** | 0.556 ± 0.004 | 0.953 ± 0.005 |
| **Enron** | **0.937 ± 0.001** | 0.817 ± 0.002 | 0.692 ± 0.065 | 0.559 ± 0.005 | 0.785 ± 0.013 | **0.918 ± 0.003** | 0.786 ± 0.013 | 0.904 ± 0.003 | **0.949 ± 0.008** | 0.901 ± 0.004 | 0.559 ± 0.024 | 0.810 ± 0.020 |
| **Flights** | 0.928 ± 0.000 | 0.786 ± 0.001 | 0.847 ± 0.000 | 0.839 ± 0.000 | 0.786 ± 0.001 | **0.983 ± 0.000** | 0.917 ± 0.011 | 0.847 ± 0.000 | **0.986 ± 0.003** | 0.913 ± 0.000 | 0.829 ± 0.004 | **0.950 ± 0.004** |
| **LastFM** | **0.877 ± 0.000** | 0.676 ± 0.001 | 0.698 ± 0.036 | 0.486 ± 0.001 | 0.676 ± 0.001 | **0.868 ± 0.003** | 0.730 ± 0.005 | 0.846 ± 0.000 | **0.914 ± 0.001** | 0.831 ± 0.000 | 0.519 ± 0.003 | 0.763 ± 0.023 |
| **MOOC** | 0.709 ± 0.001 | 0.697 ± 0.001 | 0.827 ± 0.013 | 0.563 ± 0.002 | 0.697 ± 0.001 | **0.949 ± 0.002** | 0.791 ± 0.048 | 0.616 ± 0.000 | 0.749 ± 0.046 | 0.805 ± 0.024 | 0.744 ± 0.006 | **0.881 ± 0.033** |
| **Reddit** | **0.994 ± 0.008** | 0.893 ± 0.049 | 0.955 ± 0.000 | 0.741 ± 0.067 | 0.893 ± 0.049 | **0.985 ± 0.000** | 0.949 ± 0.004 | 0.949 ± 0.004 | **0.995 ± 0.002** | **0.979 ± 0.002** | 0.964 ± 0.000 | 0.974 ± 0.000 |
| **SocialEvo** | **0.952 ± 0.001** | 0.786 ± 0.000 | **0.918 ± 0.005** | 0.499 ± 0.001 | 0.786 ± 0.000 | **0.916 ± 0.000** | 0.895 ± 0.030 | 0.858 ± 0.001 | 0.879 ± 0.032 | – | 0.684 ± 0.003 | **0.926 ± 0.009** |
| **Taobao** | 0.276 ± 0.001 | 0.270 ± 0.002 | 0.699 ± 0.000 | 0.280 ± 0.002 | 0.270 ± 0.002 | 0.757 ± 0.003 | 0.699 ± 0.002 | 0.521 ± 0.000 | **0.999 ± 0.000** | **0.862 ± 0.003** | 0.527 ± 0.024 | **0.757 ± 0.003** |
| **UCI** | **0.933 ± 0.001** | 0.921 ± 0.001 | 0.426 ± 0.040 | 0.803 ± 0.004 | 0.921 ± 0.001 | 0.918 ± 0.003 | 0.714 ± 0.011 | 0.830 ± 0.000 | **0.975 ± 0.016** | **0.970 ± 0.004** | 0.684 ± 0.008 | 0.802 ± 0.027 |
| **UNTrade** | **0.631 ± 0.002** | 0.552 ± 0.001 | **0.631 ± 0.014** | 0.488 ± 0.003 | 0.552 ± 0.001 | **0.741 ± 0.001** | **0.665 ± 0.011** | 0.584 ± 0.002 | 0.581 ± 0.096 | – | 0.596 ± 0.037 | 0.596 ± 0.017 |
| **UNVote** | 0.546 ± 0.000 | 0.517 ± 0.000 | 0.502 ± 0.020 | 0.389 ± 0.000 | 0.517 ± 0.000 | **0.593 ± 0.001** | 0.521 ± 0.008 | 0.489 ± 0.001 | **0.779 ± 0.019** | **0.588 ± 0.000** | 0.479 ± 0.003 | **0.589 ± 0.011** |
| **USLegis** | 0.621 ± 0.005 | **0.641 ± 0.002** | 0.567 ± 0.010 | 0.443 ± 0.004 | **0.641 ± 0.002** | **0.967 ± 0.003** | 0.580 ± 0.021 | 0.540 ± 0.004 | 0.531 ± 0.100 | **0.968 ± 0.002** | 0.560 ± 0.009 | **0.641 ± 0.002** |
| **Wikipedia** | 0.957 ± 0.001 | 0.910 ± 0.001 | 0.882 ± 0.003 | 0.466 ± 0.002 | 0.910 ± 0.001 | **0.989 ± 0.000** | 0.908 ± 0.004 | 0.906 ± 0.000 | **0.996 ± 0.002** | **0.988 ± 0.000** | 0.918 ± 0.002 | 0.970 ± 0.001 |
| **Transductive** | | | | | | | | | | | | |
| **CanParl** | 0.731 ± 0.000 | 0.722 ± 0.001 | 0.723 ± 0.000 | 0.717 ± 0.003 | 0.722 ± 0.001 | 0.720 ± 0.091 | **0.794 ± 0.006** | 0.723 ± 0.001 | 0.692 ± 0.072 | **0.892 ± 0.017** | 0.708 ± 0.022 | **0.758 ± 0.069** |
| **CollegeMsg** | **0.951 ± 0.002** | 0.923 ± 0.001 | 0.870 ± 0.000 | 0.875 ± 0.001 | 0.923 ± 0.001 | 0.916 ± 0.004 | 0.573 ± 0.069 | 0.870 ± 0.000 | 0.906 ± 0.012 | **0.970 ± 0.000** | 0.808 ± 0.003 | **0.923 ± 0.001** |
| **Contact** | **0.982 ± 0.000** | 0.878 ± 0.000 | **0.976 ± 0.000** | 0.794 ± 0.000 | 0.878 ± 0.000 | 0.969 ± 0.003 | 0.928 ± 0.021 | 0.938 ± 0.007 | 0.946 ± 0.021 | **0.984 ± 0.000** | 0.558 ± 0.009 | 0.977 ± 0.003 |
| **Enron** | **0.929 ± 0.001** | 0.818 ± 0.002 | 0.904 ± 0.001 | 0.690 ± 0.003 | 0.818 ± 0.002 | **0.916 ± 0.003** | 0.799 ± 0.036 | 0.829 ± 0.015 | **0.921 ± 0.003** | 0.896 ± 0.005 | 0.616 ± 0.021 | 0.862 ± 0.017 |
| **Flights** | **0.965 ± 0.000** | 0.794 ± 0.000 | 0.922 ± 0.000 | 0.907 ± 0.000 | 0.794 ± 0.000 | **0.986 ± 0.000** | 0.898 ± 0.006 | 0.945 ± 0.007 | **0.975 ± 0.006** | 0.930 ± 0.000 | 0.902 ± 0.003 | **0.979 ± 0.003** |
| **LastFM** | **0.899 ± 0.000** | 0.620 ± 0.001 | **0.895 ± 0.000** | 0.615 ± 0.000 | 0.620 ± 0.001 | **0.875 ± 0.001** | 0.679 ± 0.055 | 0.677 ± 0.059 | 0.854 ± 0.003 | 0.839 ± 0.000 | 0.509 ± 0.007 | 0.774 ± 0.026 |
| **MOOC** | 0.768 ± 0.001 | 0.741 ± 0.001 | 0.728 ± 0.001 | 0.674 ± 0.001 | 0.741 ± 0.001 | **0.946 ± 0.001** | **0.824 ± 0.032** | 0.790 ± 0.021 | 0.757 ± 0.031 | 0.807 ± 0.019 | 0.739 ± 0.006 | **0.900 ± 0.021** |
| **Reddit** | 0.976 ± 0.000 | 0.881 ± 0.000 | 0.898 ± 0.001 | 0.898 ± 0.001 | 0.881 ± 0.000 | **0.989 ± 0.000** | 0.980 ± 0.000 | 0.976 ± 0.001 | **0.985 ± 0.002** | 0.984 ± 0.002 | 0.981 ± 0.000 | **0.987 ± 0.000** |
| **SocialEvo** | **0.957 ± 0.000** | 0.783 ± 0.001 | **0.957 ± 0.000** | 0.724 ± 0.001 | 0.783 ± 0.001 | **0.952 ± 0.000** | 0.902 ± 0.003 | 0.867 ± 0.023 | 0.920 ± 0.006 | – | 0.785 ± 0.005 | **0.934 ± 0.000** |
| **Taobao** | 0.744 ± 0.004 | 0.617 ± 0.008 | 0.664 ± 0.000 | 0.752 ± 0.006 | 0.617 ± 0.008 | 0.771 ± 0.003 | 0.841 ± 0.001 | 0.840 ± 0.001 | **0.894 ± 0.002** | **0.876 ± 0.001** | 0.540 ± 0.009 | **0.865 ± 0.001** |
| **UCI** | **0.954 ± 0.001** | **0.926 ± 0.001** | 0.875 ± 0.000 | 0.877 ± 0.003 | **0.926 ± 0.001** | 0.919 ± 0.002 | 0.509 ± 0.065 | 0.879 ± 0.002 | 0.908 ± 0.012 | **0.967 ± 0.003** | 0.800 ± 0.005 | 0.888 ± 0.016 |
| **UNTrade** | **0.73 ± 0.001** | 0.554 ± 0.001 | 0.694 ± 0.001 | 0.720 ± 0.001 | 0.554 ± 0.001 | **0.751 ± 0.001** | 0.638 ± 0.003 | 0.679 ± 0.010 | **0.783 ± 0.047** | – | 0.592 ± 0.037 | 0.654 ± 0.010 |
| **UNVote** | **0.643 ± 0.001** | 0.519 ± 0.000 | 0.647 ± 0.001 | **0.643 ± 0.001** | 0.519 ± 0.000 | 0.604 ± 0.002 | 0.624 ± 0.031 | 0.652 ± 0.008 | **0.678 ± 0.041** | 0.587 ± 0.000 | 0.513 ± 0.003 | **0.718 ± 0.011** |
| **USLegis** | 0.816 ± 0.004 | 0.775 ± 0.004 | 0.771 ± 0.007 | 0.695 ± 0.009 | 0.775 ± 0.004 | **0.964 ± 0.004** | 0.743 ± 0.037 | **0.828 ± 0.002** | 0.782 ± 0.026 | **0.972 ± 0.001** | 0.774 ± 0.006 | **0.828 ± 0.002** |
| **Wikipedia** | 0.983 ± 0.000 | 0.906 ± 0.000 | 0.963 ± 0.000 | 0.672 ± 0.002 | 0.906 ± 0.000 | **0.989 ± 0.000** | 0.943 ± 0.001 | 0.951 ± 0.003 | 0.979 ± 0.003 | **0.991 ± 0.000** | 0.951 ± 0.002 | **0.985 ± 0.000** |

Table 3: Comparison of the BenchTemp (Huang et al., 2024a) leaderboard and the proposed heuristics, with the **best**, **second-best**, and **third-best** results highlighted in bold and color-coded. We observe that, in some cases, the metric becomes fully saturated, likely due to shortcomings in the sampled evaluation scenario, which tends to overestimate performance by including excessively easy negative examples.

## C Complementary Normalized Ranking Plots

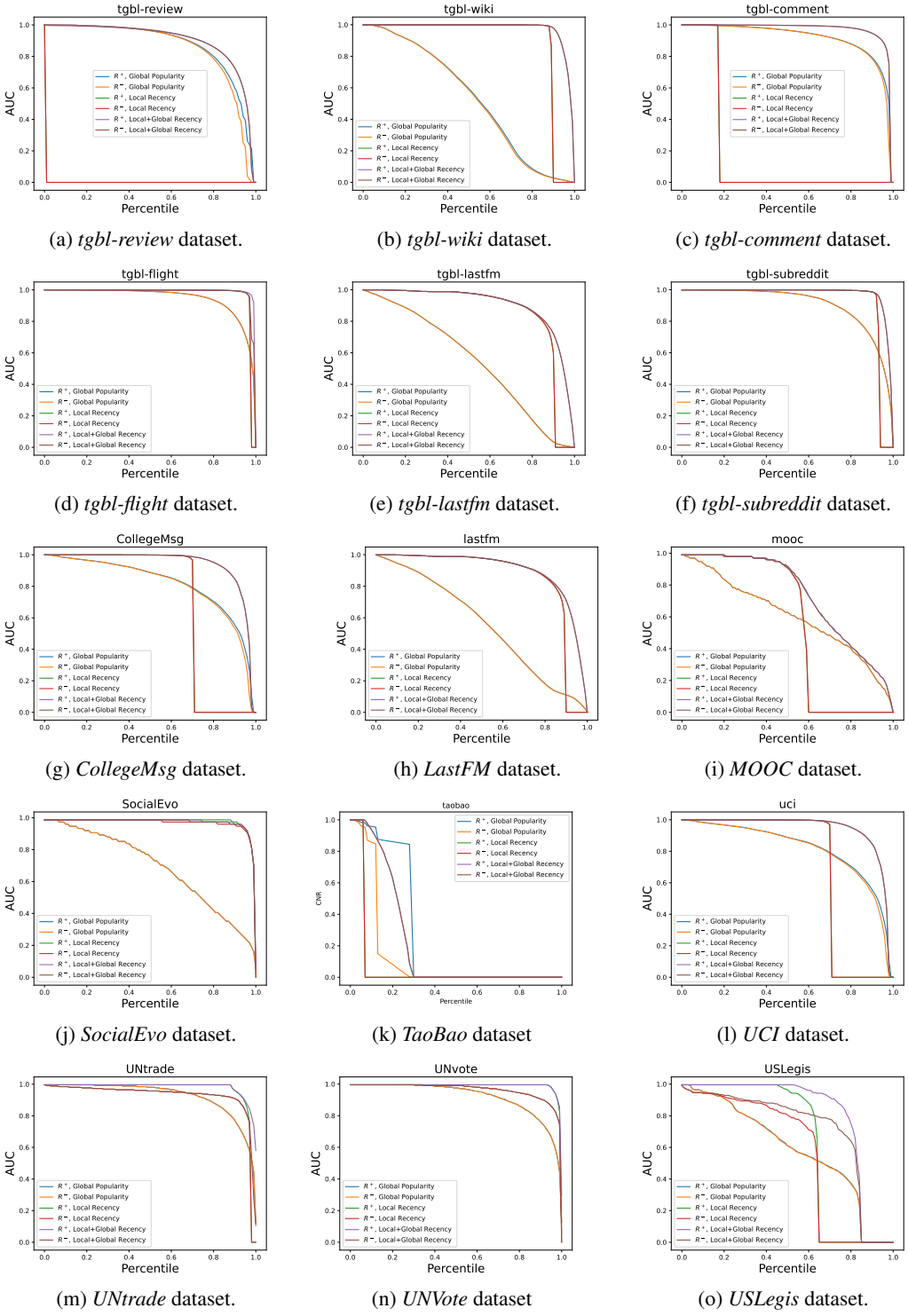
Figure 2 presents CNR plots for multiple datasets, illustrating how predicted ranks are distributed across different heuristics and highlighting key patterns in how recency and popularity influence ranking performance.

For instance, Figure 2a reveals the exceptionally poor performance of LR, largely due to approximately 98% of new edges connecting to previously unseen destination nodes. This aligns with the dataset's nature, where users typically review a product only once, making historical edges unreliable predictors of new interactions. As a result, LR struggles, as it prioritizes recently seen destinations that rarely reappear. To address this, we apply an *inverse* LR heuristic, which penalizes previously seen destinations and prioritizes unseen nodes. This adjustment better reflects the dataset's dynamics, where new interactions are more likely to involve unreviewed products. We further refine this approach by combining inverse LR with GR, significantly improving the resulting performance.

Similarly, Figure 2k shows minimal recency and popularity effects in the *TaoBao* dataset, a user-item bipartite network with a low average degree of 0.94, consistent with its structural characteristics.

Displaying both pessimistic ($R^-$) and optimistic ($R^+$) ranks is crucial for evaluating heuristics like Local Recency, which often produce ties. For example, the *USLegis* dataset (Figure 2o) exhibits a significant gap between $R^+$ and $R^-$, reflecting frequent ties. Identifying such discrepancies helps

reduce uncertainty and informs whether a single heuristic suffices or if multiple ranking strategies are needed.




(a) *tgbl-review* dataset. (b) *tgbl-wiki* dataset. (c) *tgbl-comment* dataset.

(d) *tgbl-flight* dataset. (e) *tgbl-lastfm* dataset. (f) *tgbl-subreddit* dataset.

(g) *CollegeMsg* dataset. (h) *LastFM* dataset. (i) *MOOC* dataset.

(j) *SocialEvo* dataset. (k) *TaoBao* dataset (l) *UCI* dataset.

(m) *UNtrade* dataset. (n) *UNVote* dataset (o) *USLegis* dataset.

Figure 2: Complementary Normalized Ranking Plots showing optimistic ($R^+$) and pessimistic ($R^-$) ranks for different heuristics and their combinations, for TGB (panels a-f) and BenchTemp (panels g-o) datasets. Each curve represents a method's performance across different percentiles of all edges in the dataset, illustrating how well it ranks them overall.

| | GPU | | | | | | | CPU | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Dataset** | **CAWN** | **DyRep** | **JODIE** | **NAT** | **NeurTW** | **TGAT** | **TGN** | **LR** | **GR** | **LP** | **GP** | **Combined** |
| **CanParl** | 234.65 | 2.44 | 1.96 | 3.79 | 4566.24 | 46.28 | 3.03 | 0.3 | 0.19 | 0.77 | 1.64 | 3.31 |
| **CollegeMsg** | 111.96 | 2.42 | 1.87 | 2.83 | 2097.39 | 45.8 | 2.85 | 0.82 | 0.93 | 0.95 | 1.03 | 3.65 |
| **Contact** | 12100.94 | 58.20 | 41.99 | 96.35 | 114274.38 | 1645.1 | 69.60 | — | — | — | — | — |
| **Enron** | 398.38 | 3.45 | 2.41 | 5.70 | 10896.81 | 92.24 | 4.13 | 0.8 | 0.94 | 2.84 | 3.15 | 7.45 |
| **Flights** | 12105.70 | 197.61 | 180.80 | 76.91 | 143731.99 | 1195.3 | 262.51 | 17.01 | 11.42 | 39.24 | 49.13 | 1304.46 |
| **LastFM** | 5527.12 | 39.61 | 29.42 | 51.95 | 51007.3 | 882.36 | 45.98 | 27.42 | 24.25 | 25.86 | 30.64 | 104.88 |
| **MOOC** | 1913.38 | 33.54 | 30.19 | 16.48 | 13497.27 | 256.26 | 41.48 | 6.16 | 6.6 | 6.66 | 11.21 | 29.8 |
| **Reddit** | 10896.80 | 4.10 | 3.40 | 38.60 | 5.7 | 398.3 | 92.20 | 14.39 | 12.24 | 18.21 | 18.91 | 61.76 |
| **SocialEvo** | 12292.31 | 42.57 | 27.77 | 84.72 | — | 1544.52 | 51.35 | 31.27 | 27.54 | 60.78 | 61.99 | 176.47 |
| **Taobao** | 135.04 | 38.03 | 32.17 | 4.12 | 1156.96 | 29.15 | 34.51 | 0.73 | 0.43 | 1.3 | 0.68 | 10.05 |
| **UCI** | 121.41 | 2.49 | 1.89 | 2.90 | 3801.79 | 57.49 | 2.72 | 0.82 | 0.93 | 0.95 | 1.04 | 3.67 |
| **UNTrade** | 1860.84 | 11.75 | 7.89 | 21.57 | 39402.43 | — | 14.17 | 8.25 | 5.24 | 15.22 | 21.16 | 34.92 |
| **UNVote** | 6414.90 | 25.53 | 22.31 | 40.51 | 88939.57 | 686.15 | 28.97 | 17.57 | 11.13 | 32.76 | 45.64 | 67.11 |
| **USLegis** | 220.41 | 2.73 | 2.15 | 3.16 | 3208.23 | 36.69 | 2.93 | 0.33 | 0.27 | 0.73 | 1.35 | 3.21 |
| **Wikipedia** | 270.83 | 10.02 | 9.37 | 6.15 | 4388.59 | 109.24 | 12.17 | 2.48 | 2.42 | 3.63 | 2.63 | 10.57 |

Table 4: Epoch runtimes for various models, measured in seconds.