Base3: a simple interpolation-based ensemble method for robust dynamic link prediction

Emma Kondrup
emma.kondrup@mila.quebec

ABSTRACT

Dynamic link prediction remains a central challenge in temporal graph learning, particularly in designing models that are both interpretable and effective. Existing approaches often rely on complex neural architectures, which are computationally intensive and difficult to interpret. In this work, we build on the strong recurrence-based foundation of the EdgeBank baseline [16], by supplementing it with inductive capabilities. We do so by leveraging the predictive power of non-learnable signals from two perspectives that complement EdgeBank's historical edge recurrence: global node popularity, as introduced in the PopTrack [4] model, and co-occurrence patterns through our proposed module, **t-CoMem**. t-CoMem is a lightweight memory module that tracks temporal co-occurrence patterns and neighborhood activity. Building on this, we introduce **Base3**, an interpolation-based model that fuses EdgeBank, PopTrack, and t-CoMem into a unified scoring framework. This combination effectively bridges local and global temporal dynamics – repetition, popularity, and context – without relying on training. Evaluated on the Temporal Graph Benchmark, Base3 achieves performance competitive with state-of-the-art deep models, even outperforming them on some datasets. Importantly, it considerably improves on existing baselines' performance under more realistic and challenging negative sampling strategies – offering a simple yet robust alternative for temporal graph learning.

The code used in this work is available here.

KEYWORDS

Dynamic Link Prediction, Temporal Graphs, Graph Machine Learning, Model Evaluation, Baseline

ACM Reference Format:
Emma Kondrup. 2025. Base3: a simple interpolation-based ensemble method for robust dynamic link prediction. In *Proceedings of KDD (TGL Workshop, KDD).* ACM, New York, NY, USA, 11 pages.

1 INTRODUCTION

Many real-world networks (i.e., social and financial platforms, or communication logs) are dynamic by nature and evolve continuously over time. While static graph-based models have achieved notable success in capturing structural dependencies [11, 20], they fall short in representing the temporal evolution of interactions. This gap has spurred the development of several temporal graph learning methods [10], with many tackling graph-based tasks with dynamic information [16]. However, many of these methods rely on deep neural architectures that require extensive message passing, large-scale training, and careful tuning [17, 19, 21]. Their high computational cost often makes them impractical for real-world deployment, and their complexity can considerably limit interpretability [4].

In this work, we challenge the notion that such complexity is necessary. We propose lightweight, training-free alternatives that exploit simple yet powerful temporal signals, namely edge recurrence and node popularity. Building on the success of two recent non-learnable baselines – EdgeBank, which memorizes past edges [16], and PopTrack, which models temporal popularity [4] – we introduce **t-CoMem**, a memory-based module that captures co-occurrence and neighborhood-level activity over time. We further present **Base3**, an interpolation-based model that combines EdgeBank, PopTrack, and t-CoMem into a unified scoring framework.

Despite their simplicity, our models perform competitively with state-of-the-art deep learning methods on the Temporal Graph Benchmark (TGB) [8]. Notably, they show exceptional robustness across challenging evaluation settings, including historical and inductive negative sampling – scenarios where existing models often degrade. This demonstrates that carefully designed non-learnable models can offer not only interpretability and efficiency but also strong generalization in realistic dynamic graph tasks.

2 RELATED WORK

Heuristic Approaches for Link Prediction. Before the rise of neural models, link prediction in graphs was typically approached using heuristic-based methods. Some of these remain strong, interpretable baselines today [12]. These heuristics exploit simple topological signals to estimate the likelihood of a link forming between two nodes. Among the most well-known are the *Common Neighbors* and *Preferential Attachment* measures. While these are limited in performance, especially in complex settings, they are quite straightforward in respect to both not requiring any training and offering interpretable insights.

An important principle behind many link prediction heuristic approaches is that of *Triadic Closure*, which suggests that if two nodes share a common neighbor, they are more likely to, themselves, form a direct connection [6]. It reflects tendencies toward triangle formation in real-world networks, especially social networks, which often exhibit high levels of triadic closure.

TGL Workshop, KDD, 2025, Toronto, ON

Building on this, the *Common Neighbors* score measures the size of the intersection between the neighbor sets of two nodes:

$$\mathrm{CN}(u, v) = |\Gamma(u) \cap \Gamma(v)| \tag{1}$$

where $\Gamma(x)$ denotes the set of neighbors of node x. This heuristic is particularly effective in networks where triadic closure is common [12]. Another widely used measure is *Preferential Attachment*, which is grounded in generative network theory. It assumes that high-degree nodes are more likely to form new links—a phenomenon often described as "the rich get richer". Its formulation is:

$$\mathrm{PA}(u, v) = |\Gamma(u)| \cdot |\Gamma(v)| \tag{2}$$

This heuristic is especially relevant in scale-free networks, such as citation graphs or web data [1]. Along with the *Adamic Adar Index*, defined as

$$AA(u, v) = \sum_{v \in \Gamma(u) \cap \Gamma(v)} \frac{1}{\log |\Gamma(w)|} \tag{3}$$

and the *Resource Allocation Index*, defined as

$$RA(u, v) = \sum_{w \in \Gamma(u) \cap \Gamma(v)} = \frac{1}{|\Gamma(w)|} \tag{4}$$

these are shown to reach results comparable with state-of-the-art methods, while offering more interpretability [3, 5]. They also offer insight into the structural biases that more complex models aim to learn or surpass [12, 13].

Static Graph Neural Networks (GNNs) have become foundational tools for learning on relational data. In static graphs, where the node and edge sets remain fixed, GNNs learn by aggregating and transforming features from local neighborhoods [11]. Their inherent ability to model natural dependencies between entities in the graph makes them particularly effective at capturing local structural patterns. Variants of the original GNN architecture quickly emerged, notably incorporating attention mechanisms [20] which allowed the model to learn dynamic weighting of neighbors based on their relative importance and thus moved beyond uniform aggregation. GraphSAGE, another widely adopted variant, introduced neighborhood sampling for inductive representation learning, enabling generalization to unseen nodes in large-scale static graphs [7].

Despite their usefulness, these models remain assuming a fixed topology, which limits their applicability to real-world domains such as communication networks, transportation or web data, where interactions and relationships are inherently dynamic. These evolving structures call for models that can adapt to temporal changes in the graph, motivating research in dynamic and temporal GNNs.

Temporal Graph Learning focuses on modeling spatial and temporal dependencies in evolving networks. Following the taxonomy in [10], methods are broadly divided into Discrete-Time Dynamic Graphs (DTDGs) and Continuous-Time Dynamic Graphs (CTDGs), depending on how they represent temporal evolution.

DTDG methods represent temporal dynamics using a sequence of graph snapshots sampled at fixed intervals. This discretized approach enables the reuse of static GNNs in conjunction with recurrent or temporal modules to capture historical patterns over time [15, 18]. While DTDGs offer computational efficiency and a straightforward temporal abstraction, they often struggle to capture fine-grained or asynchronous event dynamics, and may smooth over important temporal details [19]. Recent work, such as the recently-proposed Unified Temporal Graph [9], seeks to bridge this gap by adapting snapshot-based models to handle irregular event streams.

In contrast, *CTDG methods* capture interactions at precise timestamps, treating the graph as an asynchronous sequence of interactions. This finer granularity better reflects the irregular, event-driven nature of many real-world systems, such as financial transactions, messaging platforms, or online user behavior [17]. Notable CTDG models include TGAT, which introduces temporal attention and time encoding [21] and TGN, which incorporates memory modules and message queues for long-term temporal context [17]. More recent models like GraphMixer [2] use parameter-efficient mixing layers to combine structural and temporal signals, achieving strong performance on large-scale dynamic graphs. DyGFormer [22] and TNCN [23] further improve temporal modeling by introducing transformer-based spatiotemporal attention and context-aware temporal convolutions, respectively. Together, these models highlight the importance of explicitly modeling temporal granularity and memory in CTDG frameworks to capture the full complexity of evolving graph data.

Despite their expressive power, however, these usually heavy and complex models require extensive training data, computational capabilities, and careful tuning.

Dynamic Link Prediction is one of the principal tasks in Temporal Graph Learning, and consists of predicting the existence of an edge between two existing nodes at a given timeframe. The added complexity of dynamic behavior and evolving communities makes this task considerably harder than static link prediction.

Negative Sampling Strategies play a crucial role in the evaluation of temporal learning models, as highlighted in [16]. They directly impact the difficulty of the prediction task and thus the interpretability of performance metrics. The most commonly used method has been *random negative sampling*, where negative edges (non-existent links) are simply uniformly sampled from the set of all possible node pairs [17, 21]. While efficient, random sampling may yield trivial negatives – node pairs that are structurally or temporally unrelated – which can inflate confidence signals and misrepresent a model's generalization ability [16].

To address these limitations, recent work has proposed more challenging and realistic alternatives. These alternatives are crucial for confidently validating temporal graph models under realistic settings. The EdgeBank framework [16], which marked an important step in highlighting these limitations, formalizes two such strategies: *inductive* and *historical* sampling. Inductive sampling evaluates a model's ability to generalize to unseen nodes by constructing negative samples that include entities not observed during training. This setting simulates real-world cold-start or deployment scenarios where models must infer relationships for entirely new entities. Historical sampling, on the other hand, draws negatives

from node pairs that have interacted in the past but are not linked in the current prediction window. These "near-positive" negatives are more ambiguous, requiring the model to distinguish between truly inactive and merely temporarily inactive links. Both strategies increase the robustness and credibility of evaluation by focusing on more realistic and informative failure cases, aligning with the broader need for standardized, challenging benchmarks in temporal graph learning [9].

EdgeBank and **PopTrack** represent two of the most simple-yet-competitive non-learnable baselines for dynamic link prediction.

EdgeBank, introduced by Poursafaei et al. [16], is grounded in the principle following which past links are likely to re-occur. It maintains a memory of all previously seen edges—its *edge bank*—and predicts future links by checking whether a candidate edge exists in this memory. EdgeBank has two forms, EdgeBank $_{\text{tw}}$ which keeps an edge bank over a recent determined time window, and EdgeBank$_\infty$ for which the edge bank spans the entire timeframe. This memorization-based approach proves highly effective in domains with strong recurrence patterns. However, EdgeBank is fundamentally non-inductive: it cannot predict links involving node pairs never observed during training, limiting its generalization to novel interactions. Despite this, EdgeBank achieves highly competitive performance, particularly in domains with recurring relationships – and has thus since become a widely-used baseline in the temporal graph learning literature [8, 16]. Existing efforts to supplement EdgeBank with inductive capabilities have consisted in incorporating it with temporal collaborative filtering, a method which, while interesting, has yet to show itself to be highly competitive in terms of performance [14].

PopTrack [4], on the other hand, builds on the principle that node popularity correlates with connectivity. It predicts a link from node u to node v if the incident node v ranks among the top-K most popular nodes at time t, based on recent interaction frequency. Popularity is tracked using a decayed count of incoming edges, emphasizing recency while retaining longer-term trends. This makes PopTrack particularly well-suited for non-stationary environments, and where temporal bursts or shifting popularity drive link formation more than repeated edge patterns. Unlike EdgeBank, PopTrack can generalize to unseen links as long as the destination node has accumulated sufficient popularity—providing a lightweight but effective form of inductive reasoning.

3 METHODS

Our proposed **t-CoMem** module and **Base3** model build on the inductive biases of the two non-learnable baselines – EdgeBank and PopTrack – by embedding them within a simple, interpretable, and training-free framework. This framework is designed with two key goals in mind: (1) to evaluate whether straightforward memory and aggregation mechanisms can match or outperform complex neural architectures, and (2) to offer viable solutions for low-resource or real-time deployment scenarios where efficiency and interpretability are paramount.

3.1 t-CoMem

t-CoMem (Temporal Co-occurrence Memory) is a non-parametric module designed to combine two main ideas: temporal co-occurrence tracking and recent popularity weighting.

t-CoMem maintains a dynamic memory by tracking how frequently node pairs co-occur within a fixed time window (set to 1,000,000 by default), capturing short- to mid-range temporal dependencies through co-appearance patterns. To enrich this signal, it incorporates a soft popularity score from PopTrack, weighting nodes by their recent activity (their popularity score) rather than relying on binary top-K membership. Unlike PopTrack, which considers only the destination node, t-CoMem also factors in the source's recent interactions—addressing a limitation highlighted in [4] and promoting more context-aware predictions.

t-CoMem Implementation Details

Hyper-parameters:

(1) Time window `tw`, determines the wanted recency period to consider;
(2) Co-occurence weight λ, determines how strongly co-occurence affects scoring.

Data Structures:

(1) A mapping from each node u to a deque of its most recent destination nodes, $\mathcal{D}[u]$, timestamped and bounded by the time window.
(2) A dictionary storing the co-occurrence count for each node pair, $C[u][v]$.

Memory updates: memory is built through batch updates. For each batch of 200 edges, for each edge (u, v, t):

- Append (t, v) to $\mathcal{D}[u]$
- $C[u][v] \leftarrow C[u][v] + 1$
- $C[v][u] \leftarrow C[v][u] + 1$

Scoring: combines neighborhood popularity and direct co-occurrence. To do so, t-CoMem:

- Retrieves all recent destinations from $\mathcal{D}[u]$ within the time window.
- For each recent neighbor n_i, increment the score by n_i's PopTrack popularity p, exponentially decayed by how recently it was observed:

$$\text{decayed\_score} = \sum_{n_i} d \cdot p_i \tag{5}$$

where $d = \exp\left(-\frac{t-t_i}{\text{tw}}\right)$. This decay introduces recency bias, making recommendations more relevant in the common context in which recent interactions carry more predictive power, and ensuring smooth forgetting which aligns with real-world patterns.

- Retrieves the co-occurrence count c between u and v, and computes its influence f as:

$$f = \lambda \cdot \frac{c}{1+c} \tag{6}$$

- Returns the combined score using:

$$\text{score}_{\text{t-CoMem}} = \frac{1}{1 + \frac{1}{\sum d \cdot p + f}} \tag{7}$$

which squashes the result to the range $[0, 1]$.

This way, recent activity is given importance while multi-hop propagation can occur through the propagation of PopTrack popularity within these recent neighbour lists. t-CoMem's design addresses key limitations of purely popularity-based or recurrency-based models like PopTrack or EdgeBank respectively, especially PopTrack's inability to condition predictions on the source node.

3.2 Base3

Base3 is an ensemble interpolation model that linearly combines the prediction scores from EdgeBank, PopTrack, and t-CoMem. Each component contributes a weighted vote to the final score, offering a hybrid prediction that balances recurrence (EdgeBank), popularity (PopTrack), and co-occurrence (t-CoMem) – thus fusing complementary inductive signals in a modular fashion. Our proposed Base3 presents itself as a strong model that outperforms a majority of existing models, learnable and non-learnable alike.

3.3 Interpolation Models

To combine the signals from EdgeBank, PopTrack, and t-CoMem, we define an interpolated score:

$$\text{score}_{\text{Base3}}(u, v, t) = \alpha \cdot s_{\text{EB}} + \beta \cdot s_{\text{PT}} + \delta \cdot s_{\text{CM}} \tag{8}$$

where:

- $s_{\text{EB}} = s_{\text{EB}}(u, v)$ = the EdgeBank score: 1 if (u, v) has been observed before, 0 otherwise.
- $s_{\text{PT}} = s_{\text{PT}}(v)$ = the PopTrack score for the destination node v: 1 if v is in the top-K nodes, 0 otherwise.
- $s_{\text{CM}} = s_{\text{CM}}(u, v, t)$ = the t-CoMem score, which depends on the source's recent interactions and the decayed popularity of intermediate neighbors.
- (α, β, δ) are interpolation weights chosen based on the interpolation strategy.

Base3 computes a weighted sum of EdgeBank, PopTrack, and t-CoMem's outputs according to an interpolation scheme. We experiment with three such schemes:

- `Uniform` assigns equal weight to each component ($\alpha = \beta = \delta = \frac{1}{3}$), assuming that EdgeBank, PopTrack, and t-CoMem are equally informative across all contexts.
- `EB_conf` weights the components based on EdgeBank confidence. This model is based on the assumption that if a potential edge that is being scored for prediction is already in the edge bank, its EdgeBank score is more significative than otherwise, and thus that repeated interactions are highly predictive when available. Considering the population process of the edge bank, this presents itself as a promising signal. If an edge exists in EdgeBank ($s_{\text{EB}} = 1$), its contribution is thus up-weighted relative to the others. A more detailed insight into the weight repartition is made available in Appendix A.
- `multi_conf` extends `EB_conf` by also factoring in the popularity of v from PopTrack. When both EdgeBank and PopTrack show strong signals (i.e., the edge exists and the destination node is in the top-K popular nodes), both their weights are increased. If neither signal is strong, t-CoMem is favored as a fallback. Likewise, more details on this model's weight formulation process is available in Appendix A.

We explore the performance of each model in our ablation studies, finding that confidence-based models achieve higher performance. Our proposed Base3 model uses `multi_conf` by default, as it is the interpolation model which most strongly compounds the three components, as well as empirically performing the best.

4 EXPERIMENTS

4.1 Datasets

We evaluate our proposed methods across the TGB benchmark, a collection of diverse benchmark datasets for robust and reproducible evaluation [8]. TGB contains a range of 5 datasets for dynamic link prediction, which vary in size and surprise, guaranteeing a realistic overview of our models' performance across different settings. Here, *surprise* refers to the metric developed by [16] which quantifies the degree of novelty in the test set of a temporal graph, relative to the training set. It is proportional to the difficulty of predicting dynamic links on the dataset, and is defined as follows:

$$\textit{surprise} = \frac{|E_{\text{test}} \setminus E_{\text{train}}|}{|E_{\text{test}}|} \tag{9}$$

A breakdown of the TGB datasets for dynamic link prediction, and their level of surprise, is put forth in Table 2.

Table 2: Overview of TGB datasets [8]

Name	#Nodes	#Edges	#Steps	Surprise
`wiki-v2`	9,227	157,474	152,757	0.108
`review-v2`	352,637	4,873,540	6,865	0.987
`coin`	638,486	22,809,486	1,295,720	0.120
`comment`	994,790	44,314,507	30,998,030	0.823
`flight`	18,143	67,169,570	1,385	0.024

4.2 Results

We report performance using the standard evaluation metric in TGB, Mean Reciprocal Rank (MRR) on both the validation and test sets for our main experiments – we also look at Area Under the Receiver Operating Characteristic curve (AUROC) scores, in our ablation studies. For each dataset, we compare our proposed models,

Table 1: MRR score comparison with learnable models, where the first, second and *third* best performances are highlighted. * denotes our contributions.

Method	tgbl-wiki-v2		tgbl-review-v2		tgbl-coin	
	Validation MRR	Test MRR	Validation MRR	Test MRR	Validation MRR	Test MRR
Base3 *	0.727	0.743	0.101	0.108	**0.754**	**0.773**
t-CoMem *	0.381	0.432	0.090	0.108	0.689	0.702
DyGFormer [22]	**0.816 ± 0.005**	**0.798 ± 0.004**	0.219 ± 0.017	0.224 ± 0.015	0.730 ± 0.002	0.752 ± 0.004
TNCN [23]	*0.731 ± 0.001*	*0.718 ± 0.001*	0.325 ± 0.003	0.377 ± 0.010	*0.740 ± 0.002*	*0.762 ± 0.004*
TGN [17]	0.435 ± 0.069	0.396 ± 0.060	0.313 ± 0.012	0.349 ± 0.020	0.607 ± 0.014	0.586 ± 0.037
GraphMixer [2]	0.113 ± 0.003	0.118 ± 0.002	**0.428 ± 0.019**	**0.521 ± 0.015**	0.721 ± 0.005	0.763 ± 0.001

Method	tgbl-comment		tgbl-flight	
	Validation MRR	Test MRR	Validation MRR	Test MRR
Base3 *	0.426	0.450	*0.809*	*0.794*
t-CoMem *	0.341	0.447	**0.846**	**0.840**
DyGFormer [22]	*0.613 ± 0.003*	*0.670 ± 0.001*	OOT	OOT
TNCN [23]	0.642 ± 0.003	0.697 ± 0.006	0.831 ± 0.003	0.820 ± 0.004
TGN [17]	0.356 ± 0.019	0.379 ± 0.021	0.731 ± 0.01	0.705 ± 0.020
GraphMixer [2]	**0.701 ± 0.010**	**0.765 ± 0.009**	OOT	OOT

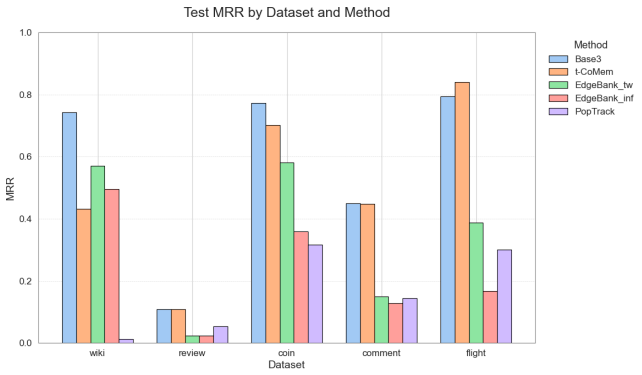

Figure 1: Test MRR by dataset for non-learnable dynamic link prediction methods (Base3 model components).

t-CoMem and **Base3**, against existing non-learnable baselines (EdgeBank and PopTrack [4, 16]) as well as state-of-the-art trainable models DyGFormer, TNCN, TGN, and GraphMixer [2, 17, 22, 23]. A key point to note about these results is that, as with original reports of EdgeBank's performance, we do not report standard deviations. This is because Base3 and its components are entirely deterministic: their predictions do not vary across runs or random seeds. Consequently, repeated executions yield identical outputs, and we therefore present only single-run results.

Figure 1 presents the MRR scores across the five datasets of the non-learnable baselines. Comparing Base3's individual components against each other strongly highlights t-CoMem as taking the lead in performance, while also putting forward the strength of compounding them together. Indeed, Base3 always outperforms its individual components by a considerable margin, except on `tgbl-flight`, on which t-CoMem itself performs best. Thus, our mixture-of-experts approach consistently outperforms existing baselines while maintaining strong levels of comprehensibility. While a non-linear learning scheme may further improve this and push Base3 to consistently outperform t-CoMem, such a method would limit interpretability. Notably, our results show:

- On `tgbl-wiki-v2`, Base3 achieves a high test MRR (0.743). Both EdgeBank and PopTrack perform more poorly on this dataset, which exhibits modest novelty (surprise = 0.108). Base3's success here stems from its ability to yield EdgeBank's recurrent pattern recognition strengths, while falling back on t-CoMem's neighborhood-aware scores when explicit memorization (EdgeBank) or popularity (PopTrack) fail.
- With the highest surprise score (0.987), the `tgbl-review-v2` dataset tests inductive generalization to unseen interactions. Base3 significantly improves over both EdgeBank and PopTrack, reaching 0.108 MRR. The improvement is largely due to t-CoMem, which compensates for EdgeBank's total failure in inductive settings and for PopTrack's limited scope. Here, Base3 benefits from its adaptive weighting strategy (`multi_conf`).
- On `tgbl-coin`, Base3 outperforms its individual components, and is closely follows by t-CoMem, while PopTrack and EdgeBank are performing considerably (more than 10%) lower. `Tgbl-coin` has a considerably low surprise rate as well (0.120), coherent with other datasets on which Base3 excels.
- `Tgbl-comment` is another high-suprise dataset (0.823), on which Base3 outperforms the two baselines by a significant margin. Notably, t-CoMem alone nearly matches Base3, suggesting that memory of co-occurrence patterns is critical in this setting. EdgeBank and PopTrack underperform due

to limited recurrence and fluctuating node popularity. This supports t-CoMem's importance and standalone strength.

- `Tgbl-flight` is the only dataset where Base3 does not outperform its best component. t-CoMem alone achieves the highest MRR among all non-learnable methods (with a score of 84%). Base3 also performs strongly, despite both EdgeBank and PopTrack's MRRs staying below 40%. We conjecture that the extreme node and edge volume, along with a very low surprise score (0.024), make recurrence highly informative, and t-CoMem's source-aware memory is a strong signal that gets diluted by PopTrack and EdgeBank.

Having established Base3 outperforms non-trained baselines, we look to determine whether it can compete with complex learnable models as well. Specifically, in Table 1, we compare our ensemble model **Base3** with the state-of-the-art DyGFormer, TNCN, TGN, and GraphMixer [2, 17, 22, 23], models which currently stand at the lead of the TGB leaderboards for dynamic link predictions. Table 1 shows that Base3 achieves consistently strong performance, outperforming diverse models across multiple settings. Notably:

- On `tgbl-wiki-v2`, Base3 ranks second overall, trailing only DyGFormer. While EdgeBank and PopTrack each perform modestly in isolation, their combination with t-CoMem in Base3 captures both recurring and contextual patterns more effectively. Remarkably, it outperforms TGN, TNCN, and GraphMixer, despite being completely training-free. This suggest complex graph analysis may be unnecessary in contexts where simple pattern recognition already excels (especially considering `wiki`'s low surprise).
- On `tgbl-review-v2` and `tgbl-comment`, the highest-surprise sets, neural models beat Base3. Base3 particularly underperforms on `tgbl-review-v2`, suggesting that in high-surprise contexts, simple pattern recognition may not be enough for robust link prediction.
- On `tgbl-coin`, Base3 delivers the highest test MRR across all methods, including state-of-the-art deep architectures. The dataset's moderate surprise score and consistent structure favor methods that blend memorization (EdgeBank) with temporal context (t-CoMem). Base3's design capitalizes on this by assigning meaningful weight to recurrence and co-occurrence without being misled by volatile popularity spikes.
- On `tgbl-flight`, not only is Base3 in the top 3 ranking, but t-CoMem itself ranks first, outperforming the TNCN and TGN models. This dataset, with its vast scale and low surprise score (0.024), is highly structured—historical recurrence is strongly predictive. In such cases, t-CoMem's ability to retain fine-grained, source-aware memory becomes dominant. Given the large size of `tgbl-flight`, however, more consuming models (DyGFormer and GraphMixer) were unable to run to completion due to computational limitations, further emphasizing the efficiency advantage of our approach. The high performance on the behalf of both t-CoMem and Base3 is especially interesting considering how low each of the baselines (EdgeBank and PopTrack) scores, comparatively.

These results confirm that combining recurrence (EdgeBank), popularity (PopTrack), and source-aware co-occurrence (t-CoMem) yields a more generalizable predictor than relying on any single heuristic alone.

4.3 Ablation Studies

To understand the role of each of Base3's hyper-parameters, we perform some ablation studies, as illustrated in Table 3. These experiments are ran with the `multi_conf` interpolation scheme. There are three hyperparameters, each of which has a considerable effect on Base3's performance:

- The first hyperparameter is the memory span. This is equivalent to the same hyperparameter introduced for EdgeBank [16], and determines how far back the memory reaches, as a percentage of the entire history. We explore with memory span values $[0.01, 0.1, 1.0]$ and find that, considerably so, a higher memory span increases performance. This is generally true, though we see a slight decrease from a 0.1 span to a 1.0 one, when K is larger. As such, we find the optimal memory span to be either 0.1 or 1.0, fixing it to 0.1 as that is optimal under optimal choices for other hyperparameters.
- The second hyperparameter is co-occurence weight, and stems from t-CoMem's logic of weighting co-occurence scores. Similarly, by trying different weights in $[0.25, 0.50, 0.75, 1.0]$, we find that a higher co-occurence weight yields higher MRR scores, and thus fix Base3's default to 1.0.
- The last hyperparameter is the K value, which stems from PopTrack's logic. In the PopTrack model, a score is positive if the destination is in the top-K most popular nodes. Interestingly, while the original authors of the model reported that the optimal K-value was 100 [4], we find that Base3's performance with $K = 100$ is much lower (about 50% so) than with $K = 1000$. We suppose this stems from the different ways in which Base3 leverages these top-K nodes and PopTrack scores themselves.
- As such, we select the optimal combination of hyperparameters and set Base3's defaults to them – specifically, we set a memory span of 0.1, a co-occurence weight of 1.0 and a K-value of 1000.

4.4 Comparing Interpolation Models

We then look at our three proposed interpolation models in more detail. Table 4 illustrates our results, looking at the `uniform`, `multi_conf` and `EB_conf` models on both the `tgbl-wiki-v2` and `tgbl-review` datasets. As can be observed, `uniform` and `multi_conf` yield the same performance on `tgbl-wiki-v2`, both being lower than `EB_conf`. This trend, however, is not reproduced in other datasets, as we empirically observed, and report for `tgbl-review`. Indeed, on `tgbl-review`, both `uniform` and `EB_conf` perform a few points lower than `multi_conf`. Considering `tgbl-review` is harder than `tgbl-wiki-v2` (it has a higher surprise), these insights are quite important, and determined

Table 3: Ablation studies of Base3 on `tgbl-wiki-v2` under varying co-occurence weights, memory spans and K values. The best performance is boldened and yields the default parameter combination used in other experiments.

Memory span	Co-occurrence weight	K	MRR_{val}	MRR_{test}
0.01	0.25	100	0.212	0.214
0.01	0.5	100	0.233	0.222
0.01	0.75	100	0.233	0.222
0.01	1.0	100	0.233	0.222
0.1	0.25	100	0.215	0.218
0.1	0.5	100	0.243	0.228
0.1	0.75	100	0.243	0.228
0.1	1.0	100	0.243	0.228
1.0	0.25	100	0.214	0.218
1.0	0.5	100	0.244	0.229
1.0	0.75	100	0.245	0.229
1.0	1.0	100	0.245	0.229
0.01	0.25	1000	0.399	0.446
0.01	0.5	1000	0.644	0.639
0.01	0.75	1000	0.649	0.644
0.01	1.0	1000	0.649	0.644
0.1	0.25	1000	0.392	0.453
0.1	0.5	1000	0.721	0.737
0.1	0.75	1000	0.727	0.743
0.1	**1.0**	**1000**	**0.727**	**0.743**
1.0	0.25	1000	0.387	0.436
1.0	0.5	1000	0.729	0.720
1.0	0.75	1000	0.736	0.727
1.0	1.0	1000	0.736	0.727

Table 4: Ablation studies of Base3 on `tgbl-wiki-v2` and `tgbl-review-v2` under varying interpolation models

Model	Memory span	Co-occurrence weight	K	tgbl-wiki-v2		tgbl-review-v2	
				MRR_{val}	MRR_{test}	MRR_{val}	MRR_{test}
Uniform	0.01	1.0	1000	0.649	0.644	0.053	0.083
	0.1	**1.0**	**1000**	**0.727**	**0.743**	**0.048**	**0.084**
	1.0	1.0	1000	0.736	0.727	0.034	0.047
Multi_conf	0.01	1.0	1000	0.649	0.644	0.102	0.108
	0.1	**1.0**	**1000**	**0.727**	**0.743**	**0.101**	**0.108**
	1.0	1.0	1000	0.736	0.727	0.101	0.107
EB_conf	0.01	1.0	1000	0.719	0.686	0.053	0.084
	0.1	**1.0**	**1000**	**0.749**	**0.752**	**0.048**	**0.084**
	1.0	1.0	1000	0.742	0.738	0.035	0.047

us setting Base3's default strategy to `multi_conf` to produce a more robust model.

4.5 Studies under different negative sampling strategies

As highlighted previously, the choice of the negative sampling strategy used is of great importance when evaluating a temporal graph model. As such, we look at Base3's performance under the three different negative sampling strategies principally used in the literature–random sampling, as well as inductive and historical.

As put forth in our ablation studies, the settings used in these experiments are set to the optimal empirical combination: the memory span is set to 0.1, the co-occurence weight to 1.0, and the K value to 1000. The interpolation model used is `multi_conf`. Here, we look at the model's Area Under the Receiver Operating Characteristic (AUROC) performance, as this is the metric used in the literature that developed inductive and historical negative sampling, namely [16].

As shown in Table 5, the existing baselines exhibit a marked drop in test MRR when the negative sampling strategy is altered—most

Table 5: AUROC performance of each model under varying negative sampling strategies on `tgbl-wiki-v2`.

Negative Sampling Strategy	Model	$\text{AUROC}_{\text{val}}$	$\text{AUROC}_{\text{test}}$
Random	**Base3**	**0.922**	**0.915**
	t-CoMem	0.914	0.909
	$\text{EdgeBank}_{\text{tw}}$	0.875	0.866
	PopTrack	0.551	0.560
Inductive	Base3	0.808	0.721
	t-CoMem	**0.923**	**0.800**
	$\text{EdgeBank}_{\text{tw}}$	0.876	0.421
	PopTrack	0.567	0.498
Historical	**Base3**	**0.797**	**0.781**
	t-CoMem	0.768	0.750
	$\text{EdgeBank}_{\text{tw}}$	0.740	0.775
	PopTrack	0.505	0.488

notably under inductive sampling. EdgeBank $_{\text{tw}}$ performs particularly poorly, achieving less than half its random sampling performance, while PopTrack suffers a nearly 10% decrease. In stark contrast, both t-CoMem and Base3 maintain robust performance across **all** sampling settings, with AUROC scores consistently above 72%. This resilience under inductive sampling highlights the effectiveness of their design and confirms the success of our core objective: endowing EdgeBank $_{\text{tw}}$ with inductive generalization through the integration of memory-based co-occurrence (t-CoMem) and popularity-aware interpolation (Base3).

5 CONCLUSION

We introduce a lightweight yet effective framework for enhancing non-learnable temporal link prediction models with inductive capabilities. By integrating co-occurrence-aware memory (t-CoMem) and popularity-driven reasoning into the EdgeBank baseline, we developed Base3, a training-free ensemble that combines recurrence, global popularity, and local temporal context into a unified scoring mechanism. Extensive evaluation across diverse datasets and under challenging negative sampling regimes demonstrates that Base3 not only outperforms traditional baselines but also rivals the performance of state-of-the-art deep learning models—without requiring training, tuning, or backpropagation. Notably, its strong performance under inductive and historical sampling confirms the success of our central objective: to enable robust generalization to unseen nodes and interactions. These results advocate for a reevaluation of complexity in temporal graph learning, suggesting that well-designed non-parametric models can offer a scalable, interpretable, and competitive alternative for dynamic link prediction in real-world applications.

6 ACKNOWLEDGEMENTS

This research was enabled in part by compute resources provided by Mila (mila.quebec).

REFERENCES

[1] Albert-László Barabási and Réka Albert. 1999. Emergence of scaling in random networks. *Science* 286, 5439 (1999), 509–512.
[2] Weilin Cong, Si Zhang, Jian Kang, Baichuan Yuan, Hao Wu, Xin Zhou, Hanghang Tong, and Mehrdad Mahdavi. 2023. Do we really need complicated model architectures for temporal networks?. In *International Conference on Learning Representations (ICLR)*.
[3] Filip Cornell, Oleg Smirnov, Gabriela Zarzar Gandler, and Lele Cao. 2025. On the Power of Heuristics in Temporal Graphs. arXiv preprint.
[4] Michal Daniluk and Jacek Dabrowski. 2024. Temporal graph models fail to capture global temporal dynamics. arXiv preprint, ICLR 2024 (withdrawn submission). openreview:9kLDrE5rsW Available at https://openreview.net/forum?id=9kLDrE5rsW.
[5] Manuel Dileo and Matteo Zignani. 2024. Link prediction heuristics for temporal graph benchmark. In *Proceedings of the European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning (ESANN 2024)*. https://www.esann.org/sites/default/files/proceedings/2024/ES2024-141.pdf
[6] David Easley and Jon Kleinberg. 2010. *Networks, Crowds, and Markets: Reasoning About a Highly Connected World*. Cambridge University Press.
[7] William L. Hamilton, Rex Ying, and Jure Leskovec. 2017. Inductive Representation Learning on Large Graphs. In *Advances in Neural Information Processing Systems (NeurIPS)*, Vol. 30. https://papers.nips.cc/paper_files/paper/2017/hash/5dd9db5e033da9c6fb5ba83c7a7ebea9-Abstract.html
[8] Shenyang Huang, Farimah Poursafaei, Jacob Danovitch, Matthias Fey, Weihua Hu, Emanuele Rossi, Jure Leskovec, Michael Bronstein, Guillaume Rabusseau, and Reihaneh Rabbany. 2023. Temporal Graph Benchmark for Machine Learning on Temporal Graphs. *Advances in Neural Information Processing Systems* (2023).
[9] Shenyang Huang, Farimah Poursafaei, Reihaneh Rabbany, Guillaume Rabusseau, and Emanuele Rossi. 2024. UTG: Towards a Unified View of Snapshot and Event Based Models for Temporal Graphs. arXiv preprint. https://arxiv.org/abs/2407.12269 arXiv:2407.12269 [cs.LG].
[10] Seyed Mehran Kazemi, Rishab Goel, Shikhar Jain, Ivan Kobyzev, Luke Sethi, Peter Forsyth, and Pascal Poupart. 2020. Representation learning for dynamic graphs: A survey. *Journal of Machine Learning Research* 21, 70 (2020), 1–73.
[11] Thomas N Kipf and Max Welling. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *International Conference on Learning Representations (ICLR)*. https://arxiv.org/abs/1609.02907
[12] David Liben-Nowell and Jon Kleinberg. 2007. The link-prediction problem for social networks. In *Journal of the American society for information science and technology*, Vol. 58. Wiley Online Library, 1019–1031.
[13] Ryan N Lichtenwalter, Jake T Lussier, and Nitesh V Chawla. 2010. New perspectives and methods in link prediction. *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining* (2010), 243–252.
[14] Shahrad Mohammadzadeh. 2024. Temporal Collaborative Filtering: Enhancing EdgeBank with Inductive Capabilities. (2024). Manuscript, not published.
[15] Amber et al. Pareja. 2020. EvolveGCN: Evolving Graph Convolutional Networks for Dynamic Graphs. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 34. 5363–5370.
[16] Farimah Poursafaei, Shenyang Huang, Kellin Pelrine, and Reihaneh Rabbany. 2022. Towards Better Evaluation for Dynamic Link Prediction. *arXiv preprint arXiv:2207.10128* (2022). https://arxiv.org/abs/2207.10128
[17] Emanuele Rossi, Ben Chambers, Rex Ying, Michael Bronstein, and Bruno Ribeiro. 2020. Temporal Graph Networks for Deep Learning on Dynamic Graphs. *arXiv preprint arXiv:2006.10637* (2020).
[18] Jo Skarding, Bogdan Gabrys, and Katarzyna Musial. 2021. Foundations and modelling of dynamic graphs. In *Proceedings of the 2021 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 1–8.

[19] Rakshit Trivedi, Mehrdad Farajtabar, Parnam Biswal, and Hongyuan Zha. 2019. DyRep: Learning Representations over Dynamic Graphs. In *International Conference on Learning Representations (ICLR)*. https://openreview.net/forum?id=HylMyhR5tm

[20] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. 2018. Graph Attention Networks. In *International Conference on Learning Representations (ICLR)*.

[21] Da Xu, Chuanwei Ruan, Evren Korpeoglu, Sushant Kumar, and Kannan Achan. 2020. Inductive Representation Learning on Temporal Graphs. In *International Conference on Learning Representations (ICLR)*.

[22] Le Yu, Leilei Sun, Bowen Du, and Weifeng Lv. 2023. Towards Better Dynamic Graph Learning: New Architecture and Unified Library. In *Advances in Neural Information Processing Systems*, Vol. 36. 67686–67700.

[23] Xiaohui Zhang, Yanbo Wang, Xiyuan Wang, and Muhan Zhang. 2024. Efficient Neural Common Neighbor for Temporal Graph Link Prediction. arXiv:2406.07926 [cs.LG]

A INTERPOLATION MODELS

Here, we provide a more detailed overview of the `EB_conf` and `multi_conf` interpolation models, specifically regarding the weighting mechanism. These schemes assign different value to the weight vector $[\alpha, \beta, \delta]$ where α is the weight given to the EdgeBank score, β that to PopTrack, and δ that to t-CoMem.

The initial weighting scheme is `uniform`, which simply linearly interpolates between the three scores by giving each of them a weight of $\frac{1}{3}$. It is the most naive and uniformly weight-assigning approach.

`EB_conf(eb_score)` is a confidence-based weighting scheme that is centered around EdgeBank. It uses one confidence signal, `eb_score` $\in [0, 1]$, which is the discrete score given by the EdgeBank module; 1 if the edge in question is present in the edgebank, and 0 otherwise. Given this signal, the scheme choose between two weight vectors: $w_{\text{conf}} = [0.5, 0.2, 0.3]$ and $w_{\text{not}} = [0.2, 0.3, 0.5]$. Essentially, the logic of these two vectors is as follows: if EdgeBank is confident, it should be the principal component relied on. We simply equate *principal* to $\frac{1}{2}$. Then, the rest of the weights is repartitioned between PopTrack and t-CoMem, with an empirically-motivated preference for t-CoMem (see the results of individual components in Figure 1). For w_{not}, the inverse logic follows: since EdgeBank is not confident, it should not be the principal component, and given t-CoMem's empirical superiority, we set that third component to be the principal one – thus achieving a weight of $\frac{1}{2}$. Likewise, the rest of the weights is shared between EdgeBank and PopTrack, with a preference for PopTrack, given the lack of confidence in EdgeBank's score.

`Multi_conf(eb_score, pop_score)` follows a similar logic, while incorporating two confidence signals: the existing `eb_score`, as well as `pt_score`, it's PopTrack analog. Recalling PopTrack's logic, it gives a score of 1.0 if the destination node being inquired is in the top-K most popular nodes, and 0 otherwise. This score is the second confidence signal `multi_conf` relies on. Similarly to `EB_conf`, the model assigns different fixed weights depending on the confidence flags. We now have four cases, as presented in Table **??**, which are based on heuristic conditional weighting that reflects the confidence signals. When both signals are reliable, EdgeBank and PopTrack are given the highest weights, while t-CoMem gets 20%. When only one of the two signals is positive, the other is downweighted: if EdgeBank is stronger, it gets 45%, while, if PopTrack is, it is given 70%. This difference is partially empirically motivated, and partially due to the consideration that t-CoMem relies on PopTrack (and thus, that strongly weighting both could be redundant). Finally, in the case where both signals are unreliable, t-CoMem is given more importance than EdgeBank. Generally, this scheme prioritizes the PopTrack weight. This choice is a consideration of PopTrack's strong performance in high-surprise datasets, most notably `tgbl-review`. Considering t-CoMem's high performance in low-surprise settings, it becomes important to combine these strengths.

Received May 2025