GRATE: Temporal Extensions for Inductive KG Foundation Models via Gated Rotary Attention

Jiaxin Pan [1] **Osama Mohammed** [1] **Daniel Hernández** [1] **Steffen Staab** [1 2]

Abstract

Knowledge graph foundation models such as ULTRA and TRIX achieve strong inductive transfer by learning relation-graph representations that generalise to unseen entities and relations. Extending this transferability to temporal knowledge graphs (TKGs) remains challenging: existing temporal models tie their parameters to dataset-specific entities, relations, or timestamps and are not designed to transfer to TKGs with disjoint vocabularies. We propose GRATE (Gated Rotary Attention for Temporal Encoding), an entity-side message function that adds no learnable parameters and encodes time through relative time differences by rotating each edge message according to its time gap to the query and applying a query-conditioned gate to select temporally relevant signals. GRATE integrates into NBFNet-style KG foundation model backbones while preserving structural transferability. Existing TKG benchmarks evaluate within shared train/test vocabularies and cannot directly test cross-dataset temporal transfer; we therefore construct GDELTINDT and WIKIINDT, fully-inductive benchmark suites with disjoint entities, relations, and timestamps spanning both interpolation and extrapolation. Across these benchmarks and held-out forecasting datasets, a single jointly pretrained GRATE checkpoint improves over the static backbone in most settings and recovers most of the performance of trained transductive models, especially when recurrence is weak.

[1]Institute for Artificial Intelligence, University of Stuttgart, Germany [2]University of Southampton, UK. Correspondence to: Jiaxin Pan <jiaxin.pan@ipvs.uni-stuttgart.de>.

Workshop on Graph Foundation Models (GFM) at the 43rd International Conference on Machine Learning (ICML), Seoul, South Korea, 2026.

1. Introduction

Recent knowledge graph (KG) foundation models (Galkin et al., 2024; Zhang et al., 2025; Du et al., 2026) have learned representations that generalise across graphs with unseen entities and relations. Many real-world facts, however, are inherently temporal: an event such as (USA, *signs agreement with*, Iran, 2015) is meaningful only when paired with a timestamp. Temporal knowledge graphs (TKGs) capture this by representing facts as (s, r, o, τ) quadruples, where s and o are the subject and object entities, r is the relation, and τ is the timestamp. Extending inductive transfer to TKGs remains challenging because existing temporal models (Zhang et al., 2022; Li et al., 2021; Sun et al., 2021) rely on dataset-specific embeddings for entities, relations, and timestamps, and must be retrained whenever new ones are introduced. In real-world scenarios such as emerging political actors, newly discovered scientists, or rapidly evolving event streams, new entities, relations, and timestamps continuously appear, making retraining costly or infeasible. A TKG foundation model that transfers without retraining is therefore a natural alternative.

Applying an inductive KG foundation model directly to a TKG leads to two key failures. First, messages from different timestamps are treated as equally aligned, ignoring their temporal distance to the query. For example, given a query (US president, *met with*, ?, 2024), supporting facts from 1974 and from 2024 are treated identically, preventing the model from distinguishing recent evidence from temporally distant facts. Second, the model aggregates all available supporting facts without selecting those relevant to the query. Even among facts at the same time, relevance varies: a 2024 meeting between the US president and a sports figure is uninformative for predicting diplomatic counterparts, whereas a same-day meeting between heads of state is highly informative. This indiscriminate aggregation introduces noise from irrelevant or redundant events.

These limitations suggest that temporal reasoning in TKGs should jointly model *temporal evolution* and *query-dependent relevance*. Temporal evolution captures how facts change relative to one another over time, while query-dependent relevance determines which supporting facts are useful for a given query. Motivated by this perspective, we

propose GRATE (Gated Rotary Attention for Temporal Encoding), which augments NBFNet-style (Zhu et al., 2021) KG foundation models with transferable temporal message passing. GRATE encodes temporal evolution by rotating (Su et al., 2024) each supporting fact's message according to its relative time gap to the query, and models query-dependent relevance through a query-conditioned gate that weights messages by their alignment with the query. Because GRATE depends only on relative time differences, it preserves the transferability of the underlying model and generalises to unseen timestamps.

To evaluate temporal reasoning in fully-inductive settings, we construct a suite of TKG benchmarks with disjoint entities, relations, and timestamps between training and inference graphs. Experiments show that GRATE improves zero-shot temporal reasoning in most settings over strong inductive backbones and recurrency-based baselines.

Our contributions are as follows: (1) We introduce GRATE, a temporal message-passing module that adds no learnable parameters and extends inductive KG foundation models to the temporal setting. (2) We construct a fully-inductive TKG benchmark suite for evaluating transfer under disjoint entities, relations, and timestamps. (3) We show that relative-time rotation and query-conditioned gating improve zero-shot temporal reasoning while preserving the efficiency and transferability of KG foundation models.

2. Related Work

2.1. Temporal Knowledge Graph Embedding

Transductive interpolation (TKG completion). TKG completion models predict missing facts at timestamps drawn from within the training time range. Early methods incorporate time as an additional component in the scoring function, including TTransE (Leblay & Chekol, 2018) and TA-DistMult (García-Durán et al., 2018). Later works such as TComplEx and TNTComplEx (Lacroix et al., 2020), TLT-ComplEx (Zhang et al., 2022), and HGE (Pan et al., 2024) extend tensor-factorisation and geometric-embedding approaches to better capture temporal dependencies. Geometric extensions in this family also include rotation-based time encodings such as TeRo (Xu et al., 2020) and Rotate3D (Gao et al., 2020), which inject a per-timestamp rotation into the scoring function.

Transductive extrapolation (TKG forecasting). TKG forecasting models predict facts at strictly future timestamps. Graph neural network approaches model the evolution of historical subgraphs to score future queries (Li et al., 2021; 2022; Han et al., 2021; Sun et al., 2021; Liang et al., 2023). Rule-based and explainable models such as TLogic (Liu et al., 2022) and xERTE (Han et al., 2020) further improve interpretability, while a unified approach (Chen et al., 2024) supports both interpolation and extrapolation under transductive vocabularies. Both lineages above lie on the transductive row of the inference grid (Section 3.2) and cannot transfer to graphs with disjoint vocabularies, which is the fully-inductive row this work targets.

2.2. KG Foundation Models

Recent work on knowledge graph reasoning has moved toward inductive, vocabulary-agnostic foundation models that generalise across graphs. Early approaches such as NBFNet (Zhu et al., 2021), GraIL (Teru et al., 2020), and INDIGO (Liu et al., 2021) introduced query-conditioned message passing that generalises to unseen entities, but assume a fixed relation vocabulary. INGRAM (Lee et al., 2023) removes this limitation by modelling relations as nodes in a relation graph, enabling generalisation to unseen relations. Building on this idea, recent KG foundation models such as ULTRA (Galkin et al., 2024) and TRIX (Zhang et al., 2025) learn transferable relational structure through message passing over relation graphs, allowing a single pretrained checkpoint to operate on graphs with disjoint entity and relation vocabularies. Concurrent work explores alternative formulations of the relation prior: KG-ICL (Cui et al., 2024) reframes inductive reasoning as in-context prompting over a query-centric prompt graph, while GraphOracle (Du et al., 2026) routes query-conditioned attention over a Relation-Dependency Graph that captures directed precedence between relations. However, these foundation models do not model time; extending them to temporal knowledge graphs without sacrificing transferability remains an open challenge, which we address in this work.

3. Problem Definition

3.1. Temporal Knowledge Graphs and Link Prediction

A temporal knowledge graph (TKG) is written $G = (V, R, T, Q)$, where each observed fact is a quadruple $(s, r, o, \tau) \in Q \subseteq V \times R \times V \times T$ with $s, o \in V$ subject and object entities, $r \in R$ a relation, and $\tau \in T$ a timestamp. We model the timestamp set T as a finite, totally ordered subset of $\mathbb{N}$, with each dataset's calendar granularity (e.g. daily, yearly) absorbed into the integer encoding.

The reasoning task studied in this work is temporal link prediction: given a query $(s, r, ?, \tau)$ or $(?, r, o, \tau)$, predict the missing entity at the specified timestamp τ. We write t_{uv} for the timestamp of an observed edge $(u \to v)$.

3.2. Inference Settings

We characterise TKG inference settings along two independent conditions. The *vocabulary condition* asks whether the entity and relation sets of the inference graph overlap

with those of the training graph; the *temporal condition* asks whether the inference timestamps fall within the training time range (*interpolation*) or strictly after it (*extrapolation*). Each condition has two values, yielding four combined settings; existing TKG benchmarks populate only the transductive (vocabulary-overlapping) row.

Vocabulary condition. We denote a training TKG as $G_{train} = (V_{train}, R_{train}, T_{train}, Q_{train})$ and an inference TKG as $G_{inf} = (V_{inf}, R_{inf}, T_{inf}, Q_{inf})$. *Transductive* settings share the entity and relation vocabularies between training and inference, $V_{train} = V_{inf}$ and $R_{train} = R_{inf}$; this is the regime in which existing TKG embedding methods learn per-entity, per-relation, and per-timestamp embeddings. *Fully-inductive* settings make the three sets disjoint,

$$V_{train} \cap V_{inf} = \emptyset,\ R_{train} \cap R_{inf} = \emptyset,\ T_{train} \cap T_{inf} = \emptyset, \tag{1}$$

so any representation tied to a specific entity, relation, or timestamp vocabulary is, by construction, not applicable at inference time. The model is trained on a source G_{train} and applied, without fine-tuning, to a target G_{inf}.

Temporal condition. *Interpolation* tasks evaluate at inference timestamps drawn from within the training time range, $T_{inf} \subseteq [\min(T_{train}), \max(T_{train})]$, and test the model's ability to align with seen temporal context. *Extrapolation* tasks evaluate at strictly future timestamps, $\min(T_{inf}) > \max(T_{train})$, and test the ability to project beyond it.

Existing benchmarks vs. our setting. No existing TKG model is designed for fully-inductive interpolation or extrapolation, which is the aim of this paper.

4. Method: GRATE

We propose Gated Rotary Attention for Temporal Encoding (GRATE), a parameter-free temporal message function for inductive knowledge graph backbones. GRATE augments the standard message function $\phi(h_u, r_{uv})$ by incorporating relative time differences between each edge and the query, enabling temporal and query-dependent reasoning while preserving transferability. It can be integrated as a drop-in replacement in NBFNet-style models such as ULTRA and TRIX. Figure 1 illustrates the GRATE component.

4.1. NBFNet-Style KG Foundation Models

We build on inductive knowledge graph foundation models based on NBFNet-style (Zhu et al., 2021) message passing, such as ULTRA (Galkin et al., 2024) and TRIX (Zhang et al., 2025). These models perform query-conditioned reasoning by propagating information over the graph.

Given a query $(s, r, ?)$, the model initialises node representations by placing the query relation at the source node s. Message passing then proceeds layer by layer over directed edges $(u \to v)$. At each layer ℓ, each edge produces a message

$$m_{uv}^{(\ell)} = \phi\big(h_u^{(\ell-1)}, r_{uv}\big), \tag{2}$$

where $h_u^{(\ell-1)}$ is the representation of node u from the previous layer, and $r_{uv} \in \mathbb{R}^d$ is the query-conditioned embedding for edge $(u \to v)$ (we reserve the symbol r for the query relation *type* in R; r_{uv} is its corresponding edge embedding produced by Eq. (4) below). Messages from neighbouring nodes are aggregated to update the representation of node v:

$$h_v^{(\ell)} = \mathrm{AGG}\big(\{m_{uv}^{(\ell)} : u \in \mathcal{N}(v)\}\big). \tag{3}$$

In vocabulary-agnostic backbones such as ULTRA (Galkin et al., 2024) and TRIX (Zhang et al., 2025), the relation embedding r_{uv} is itself computed inductively from a relation graph G_R rather than learned per relation. The relation graph treats each relation as a node and connects pairs of relations through their fundamental head/tail interaction types (h2h, h2t, t2h, t2t). Given the query relation r, both backbones produce query-conditioned relation embeddings via message passing on G_R,

$$\{r_{uv}\}_{(u,v)\in E} = \mathrm{RelGNN}(G_R \mid r), \tag{4}$$

where ULTRA runs a single forward pass over G_R that aggregates the four interaction types into a count-based relation summary, and TRIX iteratively refines the same relation graph across layers, interleaving relation-graph updates with entity-graph message passing. In both cases r_{uv} depends only on the structural position of the relation in G_R, which enables generalisation to unseen relations at inference time.

In temporal knowledge graphs, each edge is additionally associated with a timestamp. However, standard message functions ϕ do not incorporate temporal information, and therefore treat all supporting facts as temporally aligned, regardless of when they occur relative to the query.

4.2. Rotary Relative-Time Encoding

Temporal reasoning in TKGs requires modelling how relational evidence evolves relative to the query timestamp. The contribution of a supporting edge depends not only on its structural relation to the query, but also on its temporal displacement $\Delta t_{uv} = \tau - t_{uv}$ (e.g., recent events are often more informative than distant ones). In fully-inductive settings, the model must additionally remain applicable to timestamps unseen during training. The temporal message function should therefore model temporal evolution through relative time differences while generalising continuously across time.

To satisfy these requirements, we adopt rotary position embedding (RoPE) (Su et al., 2024), originally introduced for

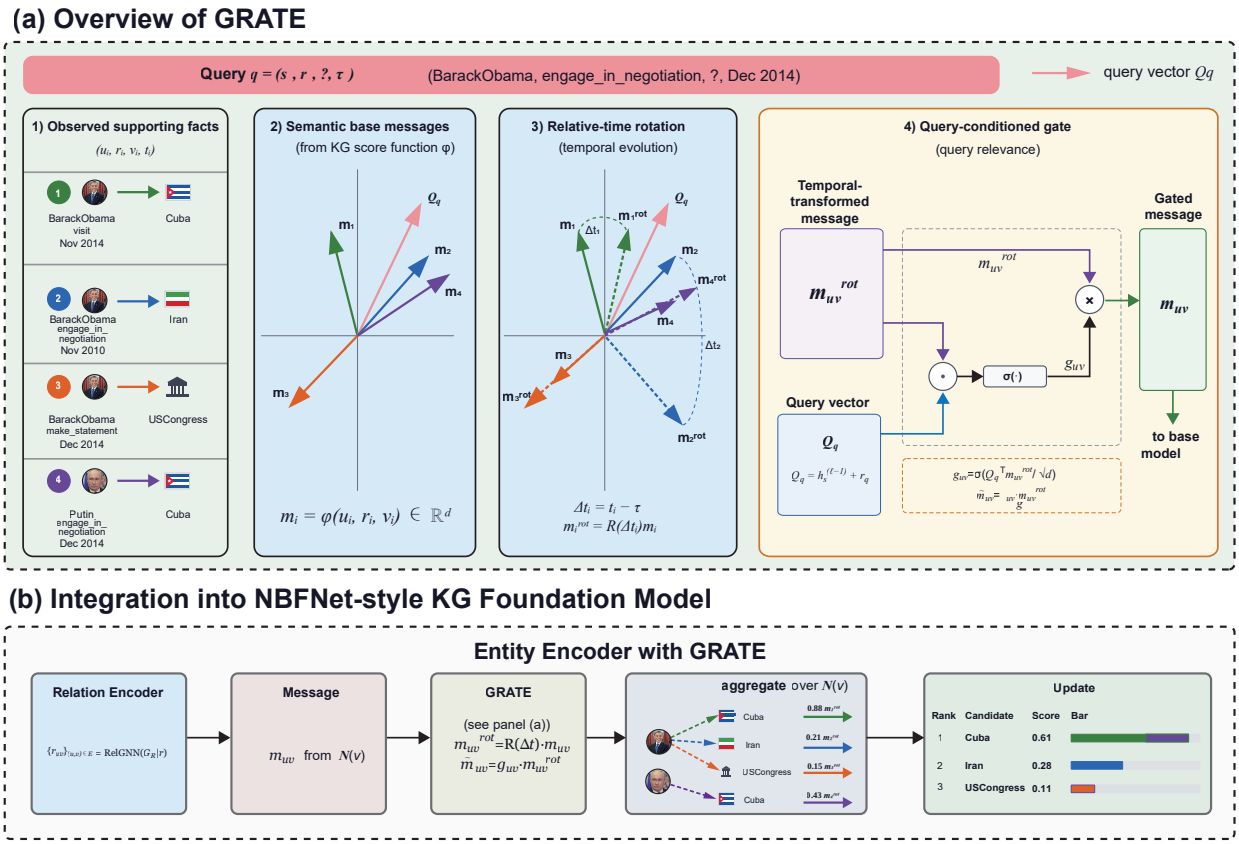

Figure 1. **GRATE mechanism and integration.** *(a)* Overview of the four processing steps for an example query (Barack Obama, *engage_in_negotiation*, ?, Dec 2014). **(1)** Observed supporting facts from the message graph. **(2)** Semantic base messages $m_{uv} = \phi(h_u, r_{uv})$ computed by the KG score function. **(3)** Relative-time rotation: each base message is rotated by $R(\Delta t_{uv})$, where $\Delta t_{uv} = \tau - t_{uv}$, encoding temporal displacement at multiple frequencies. **(4)** Query-conditioned gate: the rotated message is scored against the query state Q_q via $g = \sigma(Q_q^\top m^{\mathrm{rot}}/\sqrt{d})$, producing a gated message forwarded to the base model. *(b)* GRATE integrates into NBFNet-style KG foundation models by replacing only the entity-side message function.

positional encoding in Transformer-based language models. In the language-modelling setting, RoPE rotates token representations in a fixed frequency basis such that interactions depend on relative positional offsets. Unlike learned positional embeddings, the rotation depends only on a fixed frequency basis and varies continuously with the offset, allowing it to generalise naturally to unseen positions. These properties align closely with the requirements of fully-inductive temporal reasoning.

We use the relative time gap Δt_{uv} between an observed edge and the query timestamp as the rotation argument, applying the rotation directly to each edge message. Intuitively, the rotation transforms the message representation as a continuous function of temporal displacement, allowing relational evidence to evolve smoothly with time.

Let d denote the (even) hidden dimension of the message function and define the frequency vector

$$\omega_k = \frac{1}{10000^{2k/d}}, \qquad k = 0, 1, \ldots, d/2 - 1. \tag{5}$$

The basis spans $\omega_0 = 1$ down to $\omega_{d/2-1} \approx 10^{-4}$, so different dimension pairs encode displacement at different temporal scales. The rotation matrix $R(\delta) \in \mathbb{R}^{d\times d}$ is block-diagonal, $R(\delta) = \mathrm{diag}\big(R_0(\delta), \ldots, R_{d/2-1}(\delta)\big)$, where each 2×2 block rotates the k-th frequency pair by angle $\theta_k = \omega_k\,\delta$:

$$R_k(\delta) = \begin{pmatrix} \cos\theta_k & -\sin\theta_k \\ \sin\theta_k & \cos\theta_k \end{pmatrix}. \tag{6}$$

Partitioning a vector $x \in \mathbb{R}^d$ into consecutive pairs (x_{2k}, x_{2k+1}), the rotation acts independently on each pair:

$$\begin{aligned} \big(R(\delta)\,x\big)_{2k} &= x_{2k}\cos\theta_k - x_{2k+1}\sin\theta_k, \\ \big(R(\delta)\,x\big)_{2k+1} &= x_{2k}\sin\theta_k + x_{2k+1}\cos\theta_k. \end{aligned} \tag{7}$$

The host backbone produces a d-dimensional base message vector for each edge using the DistMult-style element-wise (Hadamard) interaction adopted by ULTRA and TRIX:

$$m_{uv} = h_u \odot r_{uv} \in \mathbb{R}^d. \tag{8}$$

GRATE then rotates this base-message vector by $R(\Delta t_{uv})$:

$$m_{uv}^{\text{rot}} = R(\Delta t_{uv})\, m_{uv}. \tag{9}$$

This construction encodes temporal evolution through relative temporal differences alone (with past and future rotating in opposite directions) and naturally generalises to unseen timestamps. Because the rotation depends only on the relative gap Δt_{uv}, the model does not rely on timestamp-specific representations, preserving transferability across disjoint temporal vocabularies.

4.3. Query-Conditioned Gating

The rotation in Equation (9) incorporates the relative time gap Δt into each message, but does not yet address a second requirement: selecting which temporally shifted edges are relevant to the current query based on their alignment with the query representation. In temporal knowledge graphs, multiple supporting facts may exist at different time offsets, and not all contribute equally to the prediction.

To model query-dependent relevance, we introduce a gating mechanism with no learnable parameters that depends jointly on the query and the rotated message. Let $h_{\text{head}(q)}^{(\ell-1)}$ denote the representation of the query head entity at layer $\ell-1$, and let r_q be the query relation embedding. We define the query state

$$Q_q = h_{\text{head}(q)}^{(\ell-1)} + r_q, \tag{10}$$

which summarises the query context. This additive form mirrors NBFNet-style initialisation, where the query relation is placed at the source node; Q_q therefore represents the same query signal that drives message propagation from the source.

Given the rotated message m_{uv}^{rot}, we compute a scalar gate

$$g_{uv} = \sigma\left(\frac{Q_q^\top\, m_{uv}^{\text{rot}}}{\sqrt{d}}\right), \tag{11}$$

where σ denotes the sigmoid function. The $1/\sqrt{d}$ scaling matches scaled dot-product attention (Vaswani et al., 2017) and stabilises gate magnitudes as d grows. The final GRATE message is

$$m_{uv}^{\text{GRATE}} = g_{uv} \cdot m_{uv}^{\text{rot}}. \tag{12}$$

The gate measures the alignment between the query and the rotated message, allowing the model to emphasise temporally relevant signals while suppressing irrelevant ones.

Unlike softmax attention, which couples edge weights within a neighbourhood, the sigmoid gate evaluates each edge independently. This matches the additive aggregation structure of NBFNet-style message passing, where supporting evidence is accumulated rather than normalised across neighbours. Independent gating is also more stable under inductive transfer, where neighbourhood-size distributions may differ substantially from training graphs. Because the gate depends only on the alignment between the rotated message $R(\Delta t_{uv})\, m$ and the query state Q_q in the shared RoPE basis, the notion of temporal relevance remains consistent across training and inference graphs without introducing timestamp-specific parameters.

Both the rotation and gating rely only on relative time differences and existing learned representations (h_u, r_{uv}, $h_{\text{head}(q)}$, r_q), so GRATE adds zero learnable parameters over the host backbone (Appendix C) while incorporating temporal and query-dependent reasoning. Algorithm 1 summarises the full per-layer forward pass.

Algorithm 1 GRATE forward pass for layer ℓ, query $(s, r, ?, \tau)$.

Require: node states $\{h_u^{(\ell-1)}\}$,
query relation embedding r_q,
edges $u \xrightarrow{r_{uv}} v$ with timestamps $\{t_{uv}\}$,
hidden dimension d (even), RoPE base b=10000.
Ensure: updated node states $\{h_v^{(\ell)}\}$.

```
ω_k ← b^{-2k/d} for k = 0, ..., d/2−1
Q_q ← h_s^{(ℓ−1)} + r_q {query state}
for all edges u —r_uv→ v with timestamp t_uv do
    Δt_uv ← τ − t_uv
    m ← h_u^{(ℓ−1)} ⊙ r_uv {base message}
    m^rot ← R(Δt_uv) m via Equation (7)
    g ← σ(Q_q^⊤ m^rot / √d) {gate}
    m_uv^GRATE ← g · m^rot
end for
for all nodes v do
    h_v^{(ℓ)} ← AGG({m_uv^GRATE : u ∈ N(v)})
end for
return {h_v^{(ℓ)}}
```

4.4. Multi-TKG Pretraining

Because GRATE applies rotary encoding over ordered time IDs, jointly training on multiple source TKGs introduces a conflict: different datasets often reuse the same local indices (e.g., starting from 0) for different timestamps or granularities. We therefore remap the timestamps of each source onto disjoint integer ranges before joint training, so no two datasets share an ID. Since GRATE depends only on relative gaps Δt, this remapping preserves per-source temporal relationships.

5. Experimental Setup

5.1. Evaluation Regimes

We evaluate GRATE under three regimes: (1) *transductive fine-tuning* on ICEWS14 and ICEWS05-15; (2) *fully-inductive transfer* on the GDELT and WIKI inductive benchmarks we construct; (3) *held-out zero-shot transfer* on ICEWS18 and YAGO under the standard single-step extrapolation protocol. Regime (2) tests the central claim of GRATE; the same jointly pretrained checkpoint is used in all three regimes.

5.2. Datasets

Pretraining sources. We jointly pretrain on ICEWS14 and ICEWS05-15 (García-Durán et al., 2018; Lautenschlager et al., 2015), two political-event TKGs from ICEWS with daily timestamps. Both use interpolation splits and are the only data the trained GRATE checkpoint sees.

Fully-inductive benchmarks. We construct GDELTINDT from GDELT (Trivedi et al., 2017) (one year of high-density daily events, April 2015–March 2016) and WIKIINDT from WIKI (Jin et al., 2020) (232 years of yearly encyclopaedic facts). Each source yields 8 V/R/T-disjoint variants (4 p_{tri} levels $\times$ 2 temporal modes), giving 16 benchmarks in total. Construction details follow in Section 5.3.

Held-out forecasting. **ICEWS18** (daily political events, January–October 2018) and **YAGO** (yearly encyclopaedic facts under a chronological split), both from Jin et al. (2020), are too small in relation and entity vocabulary to admit our V/R/T-disjoint construction. We therefore use them only for zero-shot evaluation under the standard single-step extrapolation protocol. Per-dataset statistics for all six TKGs are in Table 4 (Appendix B).

5.3. Fully-inductive benchmark construction

Split protocol. The model fits on the source TKG G_{train} and selects hyper-parameters from a held-out subset $Q_{valid} \subset Q_{train}$ that is transductive on the training graph.[1] At inference, the model is applied to the target G_{inf} without fine-tuning: an observed subset $Q_{\text{O}} \subset G_{inf}$ is the message-passing graph, and predictions are scored on a disjoint held-out $Q_{test} \subset G_{inf}$ which the model never sees during any forward pass.

[1]Two protocols exist in inductive KG evaluation: holding the validation set out from the inference graph (Lee et al., 2023), or from the training graph (Zhou et al., 2023). We adopt the latter to keep the inference graph fully unseen during training and hyper-parameter selection, eliminating leakage from the target.

Benchmark construction. We extend INGRAM Algorithm 2 (Lee et al., 2023) to quadruples with a chronological time split, substituting INGRAM's inference-side validation split with the training-side validation above. The procedure varies two orthogonal axes: $p_{\text{tri}} \in \{0.25, 0.50, 0.75, 1.00\}$ (the fraction of inference-graph edges using unseen R *and* unseen T)[2] and a temporal mode $\in \{\textbf{inter}, \textbf{extra}\}$ (random shuffle vs. chronological message/test split). Full algorithmic details are in Appendix B; per-variant sizes are in Tables 6 and 7. Test sets range from $\sim$28 K–80 K queries on GDELT to $\sim$1.2 K–5.1 K on WIKI. Naming follows INGRAM with the trailing number indicating $p_{\text{tri}} \times 100$ (e.g. WIKIINDT-50-EXTRA).

5.4. Evaluation Protocol and Baselines

Metrics. We report Mean Reciprocal Rank (MRR) and Hits@$1/3/10$ averaged over both tail prediction $(s, r, ?, \tau)$ and head prediction $(?, r, o, \tau)$, under the time-aware filter of Goel et al. (2020) which masks only the candidate entities matching known facts at the specific query timestamp τ.

Baselines per regime. (i) *Transductive*: dataset-specific TKGE methods at their published full-capacity settings, including TA-DistMult (García-Durán et al., 2018), TeRo (Xu et al., 2020), TComplEx, and TNTComplEx (Lacroix et al., 2020). These provide an in-distribution SOTA reference. (ii) *Fully-inductive*: no existing TKG model is directly applicable, as the V/R/T-disjoint construction rules out any model that learns per-entity, per-relation, or per-timestamp parameters. We therefore use the count-based recurrency predictor of Gastinger et al. (2024) as a learning-free reference point; despite requiring no learning, it reaches performance comparable to state-of-the-art trained TKG forecasting methods on standard benchmarks (Gastinger et al., 2024), making it a strong baseline for TKGs dominated by recurrent facts. (iii) *Held-out forecasting*: standard forecasting baselines RE-GCN (Li et al., 2021), xERTE (Han et al., 2020), TLogic (Liu et al., 2022), TANGO (Han et al., 2021), and TimeTraveler (Sun et al., 2021) at the unified-protocol numbers of Gastinger et al. (2023); each is trained directly on the target dataset. Across all regimes we additionally report ULTRA (Galkin et al., 2024) and TRIX (Zhang et al., 2025) retrained on the same temporal data without the GRATE module, isolating the contribution of the temporal mechanism.

5.5. Implementation

Backbone hyperparameters, optimiser, batch size, and hardware are in Appendix A; zero-shot evaluation is determinis-

[2]V/R/T-disjoint transfer is a deliberate stress test, it is most representative of rapidly-evolving domains where new entities, relations continually appear over time.

Table 1. Transductive link prediction on ICEWS14 and ICEWS05-15. ULTRA and TRIX are retrained on the same temporal pretraining mix (joint ICEWS14 + ICEWS05-15) without the GRATE module. Δ rows show the per-cell GRATE − no-GRATE improvement; % SOTA rows report the GRATE-augmented number as a percentage of the strongest baseline in that column. Best per column **bold**.

	ICEWS14			ICEWS05-15		
Method	MRR	H@1	H@10	MRR	H@1	H@10
TA-DistMult	.477	.363	.686	.474	.346	.728
TeRo	.562	.468	.732	.586	.469	.795
TComplEx	.610	**.530**	.770	.660	**.590**	.800
TNTComplEx	**.620**	.520	.760	**.670**	**.590**	.810
ULTRA	.495	.373	.731	.451	.316	.720
ULTRA + GRATE	**.620**	.520	**.793**	.621	.500	.819
Δ (ULTRA)	+.125	+.147	+.062	+.170	+.184	+.099
% SOTA	100.0%	98.1%	103.0%	92.7%	84.7%	101.1%
TRIX	.486	.366	.719	.456	.322	.722
TRIX + GRATE	.609	.507	**.793**	.617	.501	**.830**
Δ (TRIX)	+.123	+.141	+.074	+.161	+.179	+.108
% SOTA	98.2%	95.7%	103.0%	92.1%	84.9%	102.5%

tic, so we report single-run numbers throughout.[3]

6. Results

We report results across the three evaluation regimes introduced in Section 5.1: transductive in-distribution performance on the pretraining sources (Section 6.1), fully-inductive transfer on the GDELT and WIKI benchmarks we construct (Section 6.2), and zero-shot transfer to the held-out forecasting benchmarks ICEWS18 and YAGO (Section 6.3). Section 6.4 then isolates the contribution of each design choice in GRATE.

6.1. Transductive in-distribution performance

Table 1 reports transductive link prediction on ICEWS14 and ICEWS05-15 under the time-aware filter. Adding GRATE consistently improves both ULTRA and TRIX across all metrics, with gains of +0.123 to +0.170 MRR and +0.141 to +0.184 Hits@1 over the corresponding backbones without temporal modelling. Gains are largest on Hits@1, indicating that relative-time alignment primarily improves top-rank precision, and consistent across both backbones, suggesting they arise from the temporal message function itself.

GRATE adds no learnable parameters to its host backbone, yet ULTRA + GRATE (1.7×10^5 params at dim 32) and TRIX + GRATE (3.4×10^5) recover most of the performance of dataset-specific TKGE models such as TComplEx and TNTComplEx, which carry on the order of 10^6 parameters from per-entity, per-relation, and per-timestamp embeddings on the same sources (Appendix C). On ICEWS14, ULTRA + GRATE matches the best baseline in MRR and exceeds all baselines in Hits@10; on ICEWS05-15, both

[3]Code and benchmarks: https://anonymous.4open.science/r/GRATE-F431/.

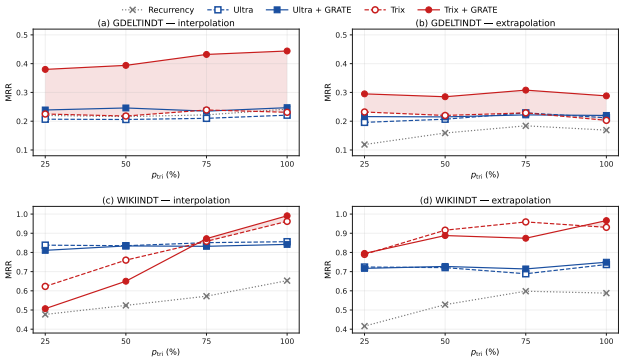

Figure 2. Zero-shot inductive transfer MRR on GDELTINDT and WIKIINDT as a function of p_{tri}. Hits@1/Hits@10 variants are in Appendix D.

GRATE variants recover over 90% of the strongest baseline MRR while remaining competitive in Hits@10. Relative-time message alignment thus captures much of the temporal signal typically modelled through dataset-specific timestamp embeddings.

6.2. Fully-inductive transfer

Figure 2 reports zero-shot transfer of a single jointly pretrained GRATE checkpoint to all 16 variants of the GDELTINDT and WIKIINDT fully-inductive benchmarks; full numerical results are in Appendix D. The two sources exhibit substantially different temporal characteristics: GDELT has low direct recurrence (DRec 2.2%, Rec 64.9%), so prediction requires integrating information across longer temporal contexts, whereas WIKI is highly recurrent (DRec 85.6%, Rec 87.0%) and many test facts can be recovered from recent history alone (Gastinger et al., 2024).

TRIX + GRATE delivers the largest gains on GDELT, improving over the corresponding static TRIX backbone by +0.155 to +0.213 MRR on interpolation and +0.063 to +0.085 MRR on extrapolation, and exceeding the recurrency baseline by +0.10 to +0.21 MRR throughout. ULTRA + GRATE also improves over static ULTRA on GDELT (+0.025 to +0.040 MRR on interpolation; roughly flat on extrapolation), but the gain is smaller than TRIX's. One possible explanation is the relation-graph distinction in Section 4.1: on dense GDELT (≈ 6.8 K edges per entity), TRIX's iterative refinement may be better suited to preserving discriminative structure for GRATE's gate than ULTRA's single-pass count summary.

On WIKI, where recurrence dominates, the static backbones already perform strongly and GRATE provides limited additional benefit at low $p_{\text{tri}} \in \{0.25, 0.50\}$; TRIX + GRATE matches or trails the static TRIX backbone by up to 0.085 MRR. At $p_{\text{tri}} = 1.0$, however, every test edge involves entirely unseen relations and timestamps and direct historical recurrence cannot serve as a shortcut. In this strict regime TRIX + GRATE again achieves the best performance

Table 2. Zero-shot transfer to ICEWS18 and YAGO. Forecasting baselines are taken from Gastinger et al. (2023) under the single-step extrapolation protocol. Δ and % SOTA rows follow the same convention as Table 1. Best per column **bold**.

	ICEWS18			YAGO		
Method	MRR	H@1	H@10	MRR	H@1	H@10
RE-GCN	**.326**	**.224**	**.526**	.822	.787	.885
xERTE	.292	.209	.463	.873	.842	.912
TLogic	.296	.204	.481	.765	.740	.792
TANGO	.284	.191	.463	.624	.590	.678
TimeTraveler	.291	.213	.439	**.877**	**.846**	.912
ULTRA	.180	.092	.369	.726	.647	.872
ULTRA + GRATE	.228	.128	.444	.792	.736	.886
Δ (ULTRA)	+.048	+.036	+.075	+.066	+.089	+.014
% SOTA	69.9%	57.1%	84.4%	90.3%	87.0%	97.1%
TRIX	.274	.170	.484	.800	.740	**.916**
TRIX + GRATE	.273	.179	.461	.858	.822	.905
Δ (TRIX)	−.001	+.009	−.023	+.058	+.082	−.011
% SOTA	83.7%	79.9%	87.6%	97.8%	97.2%	99.2%

on WIKIINDT-100 in both inter (+0.029 MRR) and extra (+0.035 MRR), indicating that relative-time alignment becomes most valuable when prediction cannot rely on memorised temporal patterns alone.

6.3. Zero-shot transfer to held-out benchmarks

Table 2 reports zero-shot evaluation on ICEWS18 and YAGO under the single-step extrapolation protocol. Prior forecasting methods (Li et al., 2021; Han et al., 2020; Liu et al., 2022; Han et al., 2021; Sun et al., 2021) are trained separately on each target; GRATE is evaluated using the same pretrained checkpoint without fine-tuning, with baseline numbers from Gastinger et al. (2023). ICEWS18 is close to the pretraining distribution (daily political events), while YAGO differs substantially in domain and temporal granularity (yearly encyclopaedic facts).

GRATE transfers strongly to YAGO despite this domain shift. TRIX + GRATE reaches 97.8% / 97.2% / 99.2% of the best directly-trained baseline in MRR / Hits@1 / Hits@10. The temporal and structural patterns learned during pretraining thus generalise effectively across domains and temporal granularities, even without exposure to the target.

On ICEWS18, the gap to fully trained forecasting models is larger but zero-shot performance remains competitive: TRIX + GRATE reaches 83.7% / 79.9% / 87.6% of the best baseline in MRR / Hits@1 / Hits@10, and ULTRA + GRATE reaches 69.9% / 57.1% / 84.4%. The contribution of GRATE itself differs across backbones here: ULTRA + GRATE improves substantially over the static ULTRA backbone (Δ MRR +0.048, Δ Hits@10 +0.075), while TRIX + GRATE adds little (Δ MRR −0.001), indicating that the static TRIX backbone already performs competitively without temporal modelling on ICEWS18. Together with the YAGO trend, these observations are consistent with GRATE's contribution depending not only on recurrence statistics (Section 6.2) but also on the gap between pretraining and target distributions.

Table 3. Component ablation of GRATE on TRIX. The top group compares temporal-encoding alternatives on top of DistMult; the bottom group adds gating components on top of rotation. GDELT-100I = GDELTINDT-100-INTER (zero-shot).

Configuration	ICEWS14 (transd. MRR)	GDELT-100I (zs MRR)
TRIX (distmult)	.486	.231
+ sincos modulation	.560	.320
+ rotation	.589	.377
rotation + rel. gate	.603	.410
rotation + query gate (GRATE)	**.609**	**.444**

6.4. Component ablation

Table 3 isolates the contribution of each GRATE component by incrementally augmenting the TRIX backbone. All variants share the same architecture and training setup, differing only in the entity-side message function.

Replacing the static DistMult message with its rotated form $R(\Delta t)$ yields gains of +0.103 MRR on ICEWS14 and +0.146 MRR on GDELTINDT-100-INTER without adding parameters, showing that most of the temporal signal can be captured by relative-time alignment alone. Rotation also outperforms an element-wise sinusoidal alternative $m \odot \mathrm{SinCos}(\Delta t)$ by 0.029 / 0.057 MRR, since rotation preserves the message norm whereas sinusoidal modulation rescales each dimension independently. Adding a relation-conditioned gate on top of rotation yields further gains of +0.014 MRR transductively and +0.033 MRR under fully-inductive evaluation; the effect is larger in the zero-shot setting, where structural and recurrent cues are weaker. Switching to query-conditioned gating yields modest in-distribution gains (+0.006 MRR) and larger fully-inductive gains (+0.034 MRR), confirming the value of query-aware edge selection.

7. Conclusion

We introduced GRATE, a parameter-free message function that lifts inductive KG foundation models to the temporal setting via relative-time rotation and query-conditioned gating. Across three evaluation regimes, GRATE improves performance in most settings, with the largest gains where recurrence and dataset-specific patterns are unavailable, and its impact scales with the richness of the host backbone's relation graph.

Acknowledgements

We gratefully acknowledge computing time on the HoreKa supercomputer at the National High-Performance Computing Center at KIT (NHR@KIT), jointly funded by the German Federal Ministry of Education and Research (BMBF), the Ministry of Science, Research, and the Arts of Baden-Württemberg, and the German Research Foundation (DFG).

Jiaxin Pan and Osama Mohammed acknowledge funding from the EU Chips Joint Undertaking (GA 101140087, SMARTY) and from the BMBF sub-project 16MEE0444. Daniel Hernández is funded by the German Research Foundation (DFG) – SFB 1574 – 471687386.

References

Chen, K., Wang, Y., Li, Y., Li, A., Yu, H., and Song, X. A unified temporal knowledge graph reasoning model towards interpolation and extrapolation. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 117–132, 2024.

Cui, Y., Sun, Z., and Hu, W. A prompt-based knowledge graph foundation model for universal in-context reasoning. In *NeurIPS*, 2024.

Du, E., Liu, S., and Zhang, Y. Graphoracle: Efficient fully-inductive knowledge graph reasoning via relation-dependency graphs. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 19055–19063, 2026.

Galkin, M., Yuan, X., Mostafa, H., Tang, J., and Zhu, Z. Towards foundation models for knowledge graph reasoning. In *The Twelfth International Conference on Learning Representations*, 2024.

Gao, C., Sun, C., Shan, L., Lin, L., and Wang, M. Rotate3d: Representing relations as rotations in three-dimensional space for knowledge graph embedding. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, pp. 385–394, 2020.

García-Durán, A., Dumančić, S., and Niepert, M. Learning sequence encoders for temporal knowledge graph completion. In Riloff, E., Chiang, D., Hockenmaier, J., and Tsujii, J. (eds.), *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 4816–4821, Brussels, Belgium, October-November 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-1516. URL https://aclanthology.org/D18-1516/.

Gastinger, J., Sztyler, T., Sharma, L., Schuelke, A., and Stuckenschmidt, H. Comparing apples and oranges? on the evaluation of methods for temporal knowledge graph forecasting. In *Joint European conference on machine learning and knowledge discovery in databases*, pp. 533–549. Springer, 2023.

Gastinger, J., Meilicke, C., Errica, F., Sztyler, T., Schülke, A., and Stuckenschmidt, H. History repeats itself: A baseline for temporal knowledge graph forecasting. In *IJCAI*, 2024.

Goel, R., Kazemi, S. M., Brubaker, M., and Poupart, P. Diachronic embedding for temporal knowledge graph completion. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 3988–3995, 2020.

Han, Z., Chen, P., Ma, Y., and Tresp, V. Explainable subgraph reasoning for forecasting on temporal knowledge graphs. In *International conference on learning representations*, 2020.

Han, Z., Ding, Z., Ma, Y., Gu, Y., and Tresp, V. Learning neural ordinary equations for forecasting future links on temporal knowledge graphs. In *Proceedings of the 2021 conference on empirical methods in natural language processing*, pp. 8352–8364, 2021.

Jin, W., Qu, M., Jin, X., and Ren, X. Recurrent event network: Autoregressive structure inferenceover temporal knowledge graphs. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 6669–6683, 2020.

Lacroix, T., Obozinski, G., and Usunier, N. Tensor decompositions for temporal knowledge base completion. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=rke2P1BFwS.

Lautenschlager, J., Shellman, S., and Ward, M. Icews event aggregations. *Harvard Dataverse*, 3(595):28, 2015.

Leblay, J. and Chekol, M. W. Deriving validity time in knowledge graph. In *Companion proceedings of the the web conference 2018*, pp. 1771–1776, 2018.

Lee, J., Chung, C., and Whang, J. J. Ingram: Inductive knowledge graph embedding via relation graphs. In *International Conference on Machine Learning*, pp. 18796–18809. PMLR, 2023.

Li, Y., Sun, S., and Zhao, J. Tirgn: Time-guided recurrent graph network with local-global historical patterns for temporal knowledge graph reasoning. In *IJCAI*, pp. 2152–2158, 2022.

Li, Z., Jin, X., Li, W., Guan, S., Guo, J., Shen, H., Wang, Y., and Cheng, X. Temporal knowledge graph reasoning based on evolutional representation learning. In *Proceedings of the 44th international ACM SIGIR conference on research and development in information retrieval*, pp. 408–417, 2021.

Liang, K., Meng, L., Liu, M., Liu, Y., Tu, W., Wang, S., Zhou, S., and Liu, X. Learn from relational correlations and periodic events for temporal knowledge graph reasoning. In *Proceedings of the 46th international ACM SIGIR conference on research and development in information retrieval*, pp. 1559–1568, 2023.

Liu, S., Grau, B., Horrocks, I., and Kostylev, E. Indigo: Gnn-based inductive knowledge graph completion using pair-wise encoding. *Advances in Neural Information Processing Systems*, 34:2034–2045, 2021.

Liu, Y., Ma, Y., Hildebrandt, M., Joblin, M., and Tresp, V. Tlogic: Temporal logical rules for explainable link forecasting on temporal knowledge graphs. In *Proceedings of the AAAI conference on artificial intelligence*, volume 36, pp. 4120–4127, 2022.

Pan, J., Nayyeri, M., Li, Y., and Staab, S. Hge: embedding temporal knowledge graphs in a product space of heterogeneous geometric subspaces. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 8913–8920, 2024.

Su, J., Ahmed, M., Lu, Y., Pan, S., Bo, W., and Liu, Y. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063, 2024.

Sun, H., Zhong, J., Ma, Y., Han, Z., and He, K. Timetraveler: Reinforcement learning for temporal knowledge graph forecasting. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 8306–8319, 2021.

Teru, K., Denis, E., and Hamilton, W. Inductive relation prediction by subgraph reasoning. In *International conference on machine learning*, pp. 9448–9457. PMLR, 2020.

Trivedi, R., Dai, H., Wang, Y., and Song, L. Know-evolve: Deep temporal reasoning for dynamic knowledge graphs. In *international conference on machine learning*, pp. 3462–3471. PMLR, 2017.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Xu, C., Nayyeri, M., Alkhoury, F., Yazdi, H. S., and Lehmann, J. Tero: A time-aware knowledge graph embedding via temporal rotation. In *Proceedings of the 28th International Conference on Computational Linguistics*, pp. 1583–1593, 2020.

Zhang, F., Zhang, Z., Ao, X., Zhuang, F., Xu, Y., and He, Q. Along the time: Timeline-traced embedding for temporal knowledge graph completion. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*, pp. 2529–2538, 2022.

Zhang, Y., Bevilacqua, B., Galkin, M., and Ribeiro, B. Trix: A more expressive model for zero-shot domain transfer in knowledge graphs. In *The Third Learning on Graphs Conference*, 2025.

Zhou, J., Bevilacqua, B., and Ribeiro, B. A multi-task perspective for link prediction with new relation types and nodes. In *NeurIPS 2023 Workshop: New Frontiers in Graph Learning*, 2023.

Zhu, Z., Zhang, Z., Xhonneux, L.-P., and Tang, J. Neural bellman-ford networks: A general graph neural network framework for link prediction. *Advances in neural information processing systems*, 34:29476–29490, 2021.

Table 4. Source dataset statistics for the six TKGs used in this paper.

Dataset	ICEWS14	ICEWS05-15	GDELT	ICEWS18	YAGO	WIKI
Entities	7,128	10,488	500	23,033	10,623	12,554
Relations	230	251	20	256	10	24
Times	365	4,017	366	303	189	232
Train	72,826	386,962	2,735,685	373,018	161,540	539,286
Validation	8,941	46,275	341,961	45,995	19,523	67,538
Test	8,963	46,092	341,961	49,995	20,026	63,110
Granularity	Daily	Daily	15 min	Daily	Yearly	Yearly

A. Implementation and Backbone Details

Unless explicitly noted otherwise, we follow the original ULTRA (Galkin et al., 2024) and TRIX (Zhang et al., 2025) configurations and training recipes.

A.1. Training Configuration

Frequency basis. GRATE uses the standard rotary frequency vector $\omega_k = 1/10000^{2k/d}$ with $k = 0, 1, \ldots, d/2 - 1$. We keep the RoPE base at 10000 in all experiments and do not tune it.

Backbones. ULTRA and TRIX use $L = 6$ message-passing layers, PNA aggregation, and hidden dimension $d = 32$ (Galkin et al., 2024). The `rspmm` CUDA kernel (Galkin et al., 2024) is unmodified; GRATE operates entirely inside the per-edge message function.

A.2. Backbone Integration

GRATE integrates into any NBFNet-style backbone as a replacement for the per-edge message function ϕ. We instantiate on two:

- **ULTRA+GRATE** replaces ULTRA's (Galkin et al., 2024) DistMult entity message with m_{uv}^{GRATE} from Equation (12). ULTRA's count-aggregated relation graph is unchanged.
- **TRIX+GRATE** performs the same swap on the TRIX backbone (Zhang et al., 2025), which uses a role-aware relation graph that distinguishes head- and tail-side incidences.

GRATE performs no surgery on the relation graph; it is purely an entity-side modification. Whatever relation features the host backbone produces, whether single-pass like ULTRA or iterative like TRIX, are consumed by the GRATE-replaced message function without modification.

A.3. Hardware and Reproducibility

All numbers reported in the main paper were obtained on NVIDIA A100 (40–80 GB) GPUs. Joint pretraining on ICEWS14 + ICEWS05-15 takes approximately two days. Code and the constructed GDELTINDT and WIKIINDT datasets are available at `https://anonymous.4open.science/r/GRATE-F431/README.md`.

B. Dataset Construction and Statistics

B.1. Source Dataset Statistics

Table 4 lists statistics for the six TKGs used in this paper. The fully-inductive splits we construct from GDELT and WIKI are reported separately in Tables 6 and 7.

B.2. Fully-Inductive Construction Procedure

Starting from a source TKG (V, R, T, Q), we extend INGRAM Algorithm 2 (Lee et al., 2023) to quadruples and add a chronological time split:

1. Restrict to the giant connected component on the entity-level graph.

Table 5. Per-source sampling parameters used to generate the GDELTINDT and WIKIINDT benchmark suites.

Source	p_R	n_{tr}	n_{inf}	hop_cap	hops_{tr}	hops_{inf}
GDELT	0.50	50	300	5	1	1
WIKI	0.25	2000	1000	20	1	2

Table 6. GDELTINDT benchmark suite ($p_R = 0.5$, $p_T = 0.3$). Edge counts in thousands.

Variant	p_{tri}	mode	$\lvert V_{\text{tr}}\rvert$	$\lvert V_{\text{inf}}\rvert$	train	valid	msg	test
-25-inter	0.25	inter	226	274	250.8	27.9	272.6	68.1
-50-inter	0.50	inter	226	274	209.9	23.3	268.4	67.1
-75-inter	0.75	inter	235	265	245.7	27.3	152.3	38.1
-100-inter	1.00	inter	230	270	220.2	24.5	123.8	30.9
-25-extra	0.25	extra	225	275	209.4	23.3	318.4	79.6
-50-extra	0.50	extra	235	265	235.1	26.1	212.6	53.2
-75-extra	0.75	extra	227	273	237.9	26.4	172.0	43.0
-100-extra	1.00	extra	233	267	239.2	26.6	114.6	28.6

2. Partition $R = R_{\text{tr}} \cup R_{\text{inf}}$ disjointly with $|R_{\text{inf}}|/|R| = p_R$.
3. Partition timestamps *chronologically*: T_{tr} contains the earliest $(1 - p_T) \cdot |T|$ timestamps, T_{inf} the remaining $p_T \cdot |T|$.
4. Sample n_{tr} seed entities and BFS-expand with a per-hop neighbour cap to obtain a candidate V_{tr}. The training edge set is $E_{\text{tr}} = \{(s, r, o, \tau) : s, o \in V_{\text{tr}}, r \in R_{\text{tr}}, \tau \in T_{\text{tr}}\}$; we take its giant component and define V_{tr} as the entities appearing in it.
5. Remove V_{tr} from the source graph; sample n_{inf} seeds from the giant component of the remainder; expand to a candidate V_{inf}.
6. Form $E_{\text{inf}}^{\text{pool}} = \{(s, r, o, \tau) : s, o \in V_{\text{inf}}\}$ and partition it into $X = \{r \in R_{\text{tr}}, \tau \in T_{\text{tr}}\}$ (seen relation *and* seen time) and $Y = \{r \in R_{\text{inf}}, \tau \in T_{\text{inf}}\}$ (strictly disjoint). Subsample to obtain $|X| : |Y| = (1 - p_{\text{tri}}) : p_{\text{tri}}$. At $p_{\text{tri}} = 1$ the inference graph is fully V/R/T-disjoint; smaller values mix in seen edges as additional supporting context.
7. Split E_{tr} into train.txt / valid.txt (90/10) and split E_{inf} into msg.txt / test.txt (80/20). The **inter** mode uses a random shuffle (msg and test span the same time window); the **extra** mode uses a chronological cut with $T_{\text{msg}} \leq T_{\text{test}}$.

B.3. Per-source Sampling Parameters

Table 5 lists the per-source sampling parameters. Both sources use $p_T = 0.3$ and split seed 0; $p_R = 0.5$ on GDELT and $p_R = 0.25$ on WIKI (lowered because WIKI's 24 relations would otherwise leave too few for both R_{tr} and R_{inf} post-GCC).

Temporal split boundaries. On GDELT, T_{tr} covers 2015-04-01 to 2015-12-12 (256 days) and T_{inf} covers 2015-12-13 to 2016-03-31 (110 days). On WIKI, T_{tr} covers the first 162 years and T_{inf} the remaining 70 years. The split is by date or year (not by quadruple count), so the inference timestamps are strictly later than every training timestamp.

B.4. GDELTINDT Statistics

Table 6 reports per-variant entity, edge, and split sizes for the eight GDELTINDT variants. All eight share the V/R/T-disjoint partition; only the inference-edge mix and the msg/test split differ.

B.5. WIKIINDT Statistics

Table 7 reports per-variant statistics for the eight WIKIINDT variants. The drop in $|V_{\text{inf}}|$ at $p_{\text{tri}} = 1$ reflects that strict V/R/T disjointness on WIKI's small 24-relation vocabulary leaves a smaller candidate inference set after the chronological time split.

C. Parameter and Complexity Analysis

Parameter count. GRATE adds no learnable parameters over the host backbone. The rotation $R(\Delta t)$ in Equation (9) is parameterised entirely by the fixed RoPE frequency vector $\omega_k = 1/10000^{2k/d}$; the gate g_{uv} in Equation (11) is a parameter-free composition of a dot product between Q_q (which reuses the layer-$\ell-1$ head state and the existing query relation embedding) and the rotated message, followed by a $1/\sqrt{d}$ scaling and a sigmoid. Table 8 reports parameter counts on the two transductive ICEWS sources at the dimension $d = 32$ used in the main paper. TKGE methods learn per-entity, per-relation, and per-timestamp embeddings, so their counts grow with $|V|$, $|R|$, and $|T|$ and vary substantially across

Table 7. WIKIINDT benchmark suite ($p_R = 0.25$, $p_T = 0.3$). Edge counts in thousands.

Variant	p_{tri}	mode	$\lvert V_{\mathrm{tr}}\rvert$	$\lvert V_{\mathrm{inf}}\rvert$	train	valid	msg	test
-25-inter	0.25	inter	722	1,089	27.7	3.1	20.3	5.1
-50-inter	0.50	inter	765	942	28.2	3.1	9.9	2.5
-75-inter	0.75	inter	743	821	30.6	3.4	5.2	1.3
-100-inter	1.00	inter	784	235	35.2	3.9	4.7	1.2
-25-extra	0.25	extra	873	1,040	34.6	3.8	19.4	4.8
-50-extra	0.50	extra	851	909	37.2	4.1	7.6	1.9
-75-extra	0.75	extra	768	843	33.4	3.7	6.3	1.6
-100-extra	1.00	extra	855	240	35.8	4.0	4.8	1.2

Table 8. Trainable parameter counts on the two transductive ICEWS sources. TKGE baselines (Lacroix et al., 2020) are dataset-specific. Backbone counts at $L = 6$, $d = 32$, PNA aggregation are dataset-independent.

Model	ICEWS14	ICEWS05-15
TComplEx	2.6×10^6	3.8×10^6
TNTComplEx	2.7×10^6	3.9×10^6
ULTRA (Galkin et al., 2024)	1.7×10^5 (dataset-independent)	
ULTRA + GRATE	1.7×10^5 (GRATE adds 0)	
TRIX (Zhang et al., 2025)	3.4×10^5 (dataset-independent)	
TRIX + GRATE	3.4×10^5 (GRATE adds 0)	

datasets. ULTRA and TRIX are vocabulary-agnostic foundation models whose counts at fixed (L, d) are dataset-independent, and GRATE adds zero on top of either.

GRATE-augmented backbones use roughly an order of magnitude fewer parameters than the TKGE baselines on ICEWS14 (1.7–3.4×10^5 vs. $\sim 2.7 \times 10^6$) and ICEWS05-15 (1.7–3.4×10^5 vs. $\sim 3.9 \times 10^6$). A parameter-free mechanism in time also has a structural advantage under fully-inductive evaluation: there is no parameter to mismatch when $T_{inf} \cap T_{train} = \emptyset$, whereas TKGE methods maintain a per-timestamp vocabulary of size $|T_{train}|$ that is by definition unusable when the inference timestamps are disjoint.

Time and space complexity. Let M denote the number of edges in the current message-passing graph, L the number of message-passing layers, and d the hidden dimension. A static NBFNet-style backbone performs $\mathcal{O}(Md)$ work per layer for the entity-side message and aggregation (Zhu et al., 2021; Galkin et al., 2024). GRATE adds, per edge, $\mathcal{O}(1)$ to compute $\Delta t_{uv} = \tau - t_{uv}$, $\mathcal{O}(d)$ to apply the pair-wise rotation $R(\Delta t_{uv})\, m$ (Equation (7)), $\mathcal{O}(d)$ for the dot product $Q_q^\top m_{uv}^{\mathrm{rot}}$ in the gate, and $\mathcal{O}(1)$ for the sigmoid and the gate multiply. Summing over edges, GRATE contributes $\mathcal{O}(Md)$ per layer, the same order as the host backbone, so the total per-query forward time is $\mathcal{O}(LMd)$, matching the static backbone asymptotically. Per-layer memory is also $\mathcal{O}(Md)$, dominated by the cos/sin rotation factors and the rotated message. The rotation and gate are simple elementwise operations that fit between the existing `rspmm` (Galkin et al., 2024) kernel calls without modification; empirically, GRATE's forward pass adds a single-digit percentage to the static backbone forward time on the same subgraph.

D. Full Numerical Results

D.1. Fully-Inductive Benchmark Tables

Table 9 reports the full numerical results underlying Figure 2, including Hits@1 and Hits@10 alongside MRR for every cell of the GDELTINDT and WIKIINDT benchmark suites.

D.2. Additional Hits@K Results

Figures 3 and 4 plot Hits@1 and Hits@10 in the same format as the MRR-based Figure 2 to make the per-cell trends easy to read.

E. Limitations and Future Work

GRATE encodes time purely through relative index differences and does not explicitly model temporal granularity or irregular intervals, which weakens its effect on coarse-grained datasets. In addition, while GRATE adds no parameters, the underlying NBFNet-style message passing remains computationally expensive on large TKGs. Combining relative-time encoding with more scalable propagation strategies, and broader pretraining mixes including GDELT or interval-style TKGs, is a natural

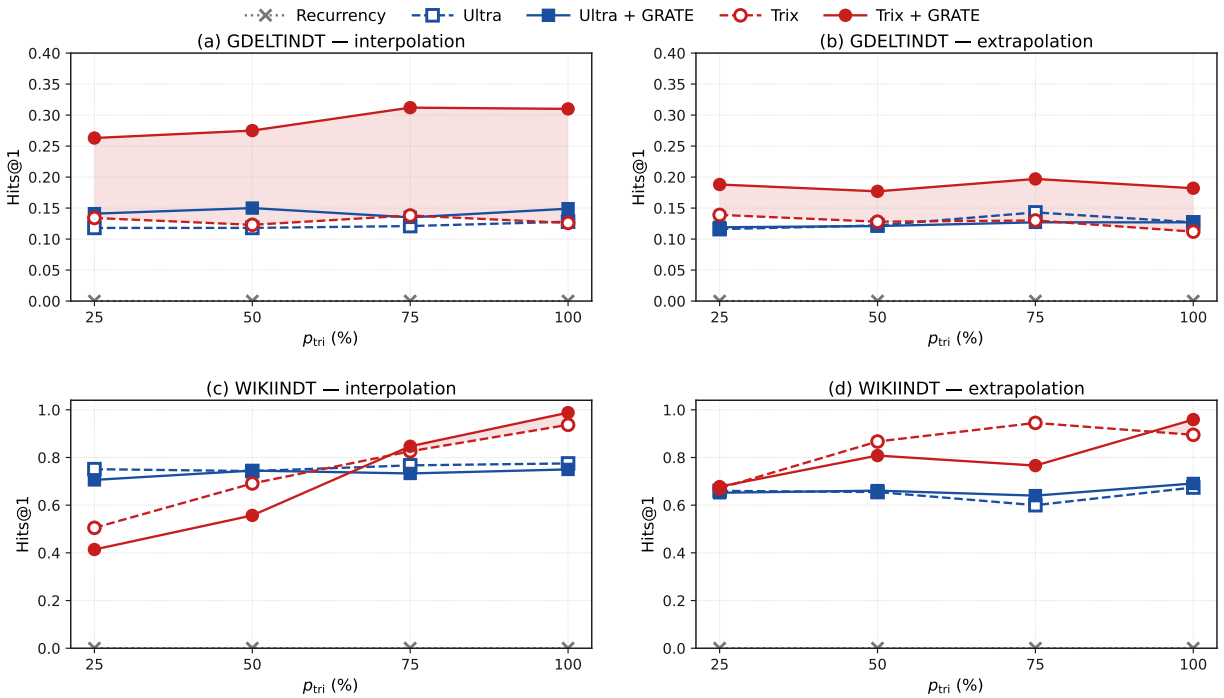

Figure 3. Hits@1 counterpart of Figure 2. The recurrency baseline sits at 0 across all panels because the count-based predictor breaks ties uniformly and never returns the gold tail at exact rank 1.

direction for future work. A complementary V-only-disjoint inductive split, in which relations and timestamps overlap with training but entities are new, would isolate the entity-vocabulary axis of transfer and is a sensible follow-up benchmark.

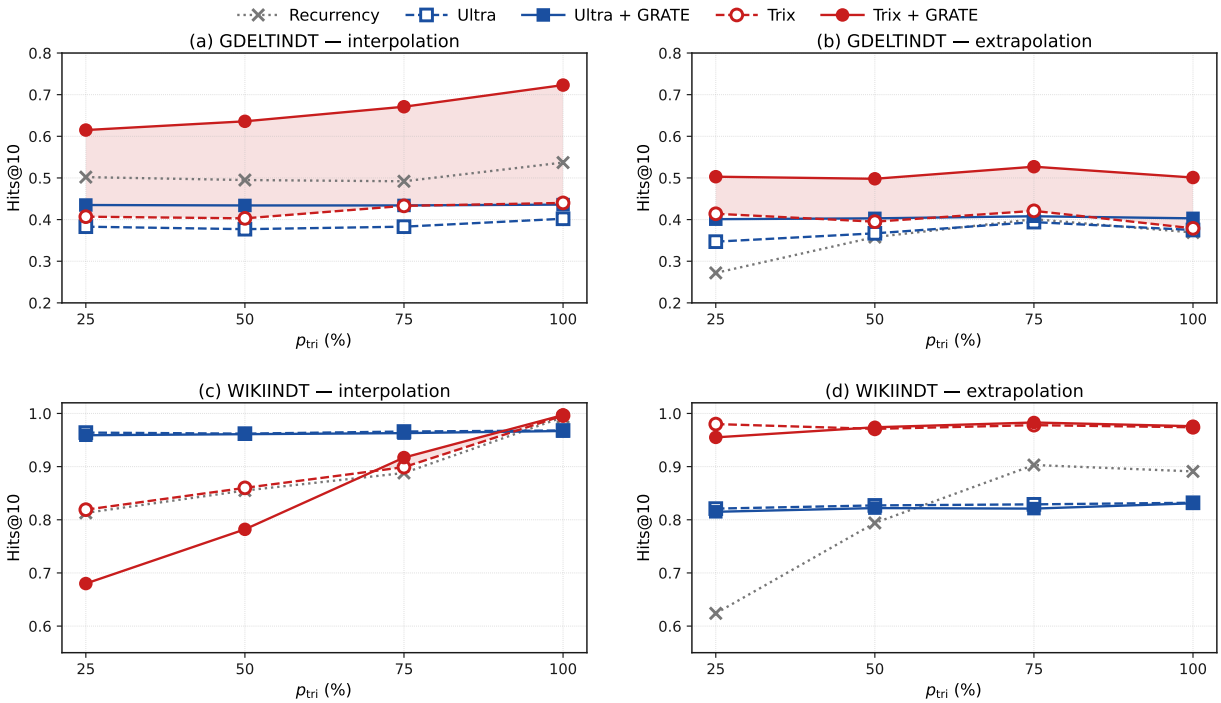

Figure 4. Hits@10 counterpart of Figure 2. On WIKIINDT-interpolation the curves saturate near the ceiling, compressing the differences visible under MRR.

Table 9. Fully-inductive zero-shot transfer to the GDELTINDT and WIKIINDT benchmark suites under the time-aware filter. Each p_{tri} column reports MRR / H@1 / H@10. Rows grouped by backbone, with Δ rows showing the GRATE − no-GRATE difference per cell. All learned methods use the same joint checkpoint pretrained on ICEWS14 + ICEWS05-15; the ULTRA and TRIX rows are retrained on the same temporal data without the GRATE module. Recurrency baseline from Gastinger et al. (2024); ULTRA from Galkin et al. (2024); TRIX from Zhang et al. (2025). Best per column **bold**.

(a) GDELTINDT:interpolation mode

	p_{25}			p_{50}			p_{75}			p_{100}		
Method	MRR	H@1	H@10	MRR	H@1	H@10	MRR	H@1	H@10	MRR	H@1	H@10
Recurrency	.220	.000	.502	.216	.000	.495	.222	.000	.492	.242	.000	.537
ULTRA	.207	.118	.383	.206	.118	.377	.210	.121	.383	.221	.128	.402
ULTRA + GRATE	.239	.141	.435	.246	.150	.434	.235	.135	.434	.247	.149	.436
Δ (ULTRA)	+.032	+.023	+.052	+.040	+.032	+.057	+.025	+.014	+.051	+.026	+.021	+.034
TRIX	.225	.134	.407	.218	.123	.403	.239	.138	.433	.231	.126	.440
TRIX + GRATE	**.380**	**.263**	**.615**	**.394**	**.275**	**.636**	**.432**	**.312**	**.671**	**.444**	**.310**	**.723**
Δ (TRIX)	+.155	+.129	+.208	+.176	+.152	+.233	+.193	+.174	+.238	+.213	+.184	+.283

(b) GDELTINDT:extrapolation mode

	p_{25}			p_{50}			p_{75}			p_{100}		
Method	MRR	H@1	H@10	MRR	H@1	H@10	MRR	H@1	H@10	MRR	H@1	H@10
Recurrency	.119	.000	.272	.159	.000	.358	.184	.000	.401	.169	.000	.369
ULTRA	.196	.116	.347	.207	.122	.367	.229	.143	.394	.212	.127	.375
ULTRA + GRATE	.216	.119	.401	.215	.121	.403	.222	.127	.408	.220	.127	.403
Δ (ULTRA)	+.020	+.003	+.054	+.008	−.001	+.036	−.007	−.016	+.014	+.008	±.000	+.028
TRIX	.232	.139	.414	.220	.128	.395	.229	.130	.421	.203	.112	.379
TRIX + GRATE	**.295**	**.188**	**.503**	**.285**	**.177**	**.498**	**.308**	**.197**	**.527**	**.288**	**.182**	**.501**
Δ (TRIX)	+.063	+.049	+.089	+.065	+.049	+.103	+.079	+.067	+.106	+.085	+.070	+.122

(c) WIKIINDT:interpolation mode

	p_{25}			p_{50}			p_{75}			p_{100}		
Method	MRR	H@1	H@10	MRR	H@1	H@10	MRR	H@1	H@10	MRR	H@1	H@10
Recurrency	.477	.000	.813	.524	.000	.855	.572	.000	.888	.653	.000	.993
ULTRA	**.838**	**.751**	**.964**	**.835**	.743	**.962**	.851	.767	**.966**	.856	.775	.968
ULTRA + GRATE	.811	.706	.959	.834	**.745**	.961	.832	.733	.963	.842	.750	.967
Δ (ULTRA)	−.027	−.045	−.005	−.001	+.002	−.001	−.019	−.034	−.003	−.014	−.025	−.001
TRIX	.623	.505	.819	.760	.691	.860	.858	.826	.899	.962	.937	.996
TRIX + GRATE	.507	.414	.680	.650	.557	.782	**.872**	**.847**	.917	**.991**	**.988**	**.997**
Δ (TRIX)	−.116	−.091	−.139	−.110	−.134	−.078	+.014	+.021	+.018	+.029	+.051	+.001

(d) WIKIINDT:extrapolation mode

	p_{25}			p_{50}			p_{75}			p_{100}		
Method	MRR	H@1	H@10	MRR	H@1	H@10	MRR	H@1	H@10	MRR	H@1	H@10
Recurrency	.416	.000	.624	.528	.000	.794	.598	.000	.903	.588	.000	.891
ULTRA	.724	.660	.821	.721	.655	.827	.689	.600	.829	.737	.674	.832
ULTRA + GRATE	.717	.652	.815	.727	.661	.822	.714	.640	.821	.749	.691	.831
Δ (ULTRA)	−.007	−.008	−.006	+.006	+.006	−.005	+.025	+.040	−.008	+.012	+.017	−.001
TRIX	.790	.671	**.980**	**.916**	**.867**	.971	**.959**	**.945**	.978	.931	.895	.974
TRIX + GRATE	**.795**	**.678**	.955	.888	.808	**.974**	.874	.766	**.983**	**.966**	**.959**	**.976**
Δ (TRIX)	+.005	+.007	−.025	−.028	−.058	+.003	−.085	−.179	+.005	+.035	+.064	+.001