# BEYOND PREDEFINED QUERIES: OPEN-DOMAIN FORECASTING ON TEMPORAL KNOWLEDGE GRAPHS

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

Forecasting future interactions and relationships has broad applications in domains such as supply chains, corporate activities, and international relations. Temporal knowledge graphs (TKGs) represent time-dependent relational facts, but conventional TKG forecasting typically predicts a missing field in a predefined future query. For example, a model may predict which company a given supplier will supply on a specified future date, while the supplier, relation, and timestamp are provided. In many applications, these fields are unknown and must be predicted together. To address this gap, we are the first to formulate the Open-Domain Forecasting problem on TKGs and propose an **O**pen-**D**omain fore**CAST**ing (**ODCAST**) framework to address it. ODCAST predicts a variable-size set of complete future relational facts from a historical TKG and a forecast horizon without predefined target queries. ODCAST uses prediction slots to form initial hypotheses and Self-Instantiated Queries to retrieve historical evidence under a shared graph-access budget. A set decoder combines neural proposals with recurrence modeling and selects complete quadruples through branch-specific thresholds, while an independent temporal trajectory expert reranks recurrent candidates. Experiments on three benchmark datasets show that ODCAST consistently outperforms the best-performing baseline in Strict F1 across all evaluated forecast horizons, achieving improvements of up to 43.15%, 110.93%, and 25.04% on ICEWS14, ICEWS05–15, and tkgl-polecat, respectively.

## 1 INTRODUCTION

Predicting what may happen in the future is important for identifying potential risks and preparing timely responses. In supply-chain monitoring, for example, forecasting acquisitions, supply changes, and emerging partnerships helps organizations assess potential disruptions and prepare responses. Yet the companies involved, their relationships, and the timing are precisely what forecasting must uncover. Temporal knowledge graphs (TKGs) represent time-dependent facts as quadruples $(s, r, o, t)$, where $s$ and $o$ denote the subject and object entities, $r$ denotes their relation, and $t$ specifies when the fact holds. Forecasting complete future facts from this history therefore helps identify what requires attention and when.

Existing TKG forecasting methods model relational evolution, exploit historical repetition, or learn temporal rules (Li et al., 2021b; Gastinger et al., 2024b; Liu et al., 2022). However, their common evaluation setting asks models to complete a *query*, i.e., a partially specified future fact. For example, Fig. 1(a) ranks candidate companies for $(\mathrm{Aster}, \mathrm{acquires}, ?, t_0 + \delta_t)$, where $t_0$ marks the end of the observed history and $\delta_t$ is the time-step duration. Ranking Cedar first completes the query, but the acquirer, relation, and time are already given. Consequently, relevant developments may remain unreported when no corresponding query is supplied. To address this gap, we formulate *Open-Domain Forecasting*, which predicts a variable-size set of complete future relational facts. Its inputs are only a historical TKG and a *forecast horizon*, i.e., how far into the future to predict. As illustrated in Fig. 1(b), the output may include Aster acquiring Cedar at $t_0 + \delta_t$ and Nova supplying Aster at $t_0 + 2\delta_t$. Neither fact is specified beforehand, so the model must infer the entities, relations, timestamps, and output count.

To enable effective forecasting of complete future facts without predefined queries, we need to address the following two significant challenges (CH).



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

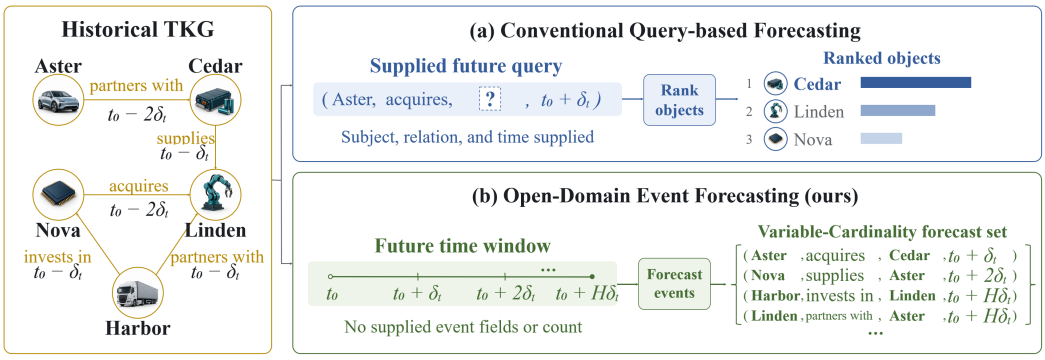



Figure 1: An example comparing the conventional TKG forecasting task with our proposed Open-Domain Forecasting task.

**CH1. How can relevant historical evidence be identified without a predefined query?** Without a supplied entity or relation, the model must form tentative predictions, i.e., *hypotheses about future facts*, to guide historical retrieval. However, reliable hypotheses require relevant evidence, while selecting that evidence depends on the hypotheses themselves. Moreover, exhaustively examining possible facts and their supporting histories is costly on large graphs. The challenge is therefore to construct informative queries under a *graph-access budget*, i.e., a limit on the number of distinct historical facts scored during retrieval.

**CH2. How can complete future facts be selected without knowing their number?** A complete forecast must specify both entities, their relation, and the future time. Repeated observations of the same relation between the same entities, termed *recurrence*, provide useful evidence. However, frequency alone cannot establish whether a fact will persist or reappear. Moreover, the number of future facts varies across windows, so a fixed output count may cause omissions or false predictions. The challenge is therefore to assess complete candidates and select a set of distinct facts without a predefined size.

To address these challenges, we propose an **O**pen-**D**omain fore**CAST**ing (**ODCAST**) framework on TKGs. First, we propose *prediction slots* to generate initial hypotheses about future facts. Next, we propose *Self-Instantiated Queries* (SIQs) from these hypotheses to retrieve relevant historical evidence. Finally, we develop a *set decoder* with recurrence modeling to select complete future facts and determine how many to report. Our main contributions are summarized as follows:

- We formulate open-domain forecasting on TKGs, requiring the prediction of complete future relational facts and their number from historical graph structure and timestamps without predefined target queries.
- We propose a forecasting framework[1] that uses predicted identities to construct historical queries and prediction uncertainty to allocate a shared access budget. It combines structured candidate scoring with recurrence-aware set selection and independent trajectory-based reranking.
- We propose an evaluation protocol for open-domain forecasting and evaluate our framework on multiple benchmark datasets. The results show that our method outperforms the best-performing baseline in Strict F1, achieving improvements of up to 43.15%, 110.93%, and 25.04% on ICEWS14, ICEWS05–15, and tkgl-polecat, respectively.

## 2 Related Work

**Temporal graph forecasting.** Existing methods model temporal dynamics and recurrence (Trivedi et al., 2017; Jin et al., 2020; Zhu et al., 2021; Li et al., 2021b) or reason over historical evidence (Han et al., 2021; Li et al., 2021a; Liu et al., 2022; Li et al., 2022b). Recent work explores diffusion (Cao et al., 2025), LLM-based forecasting (Bai et al., 2025; Yin et al., 2026), and confidence-based

[1]Our code and datasets are available in the supplementary material.

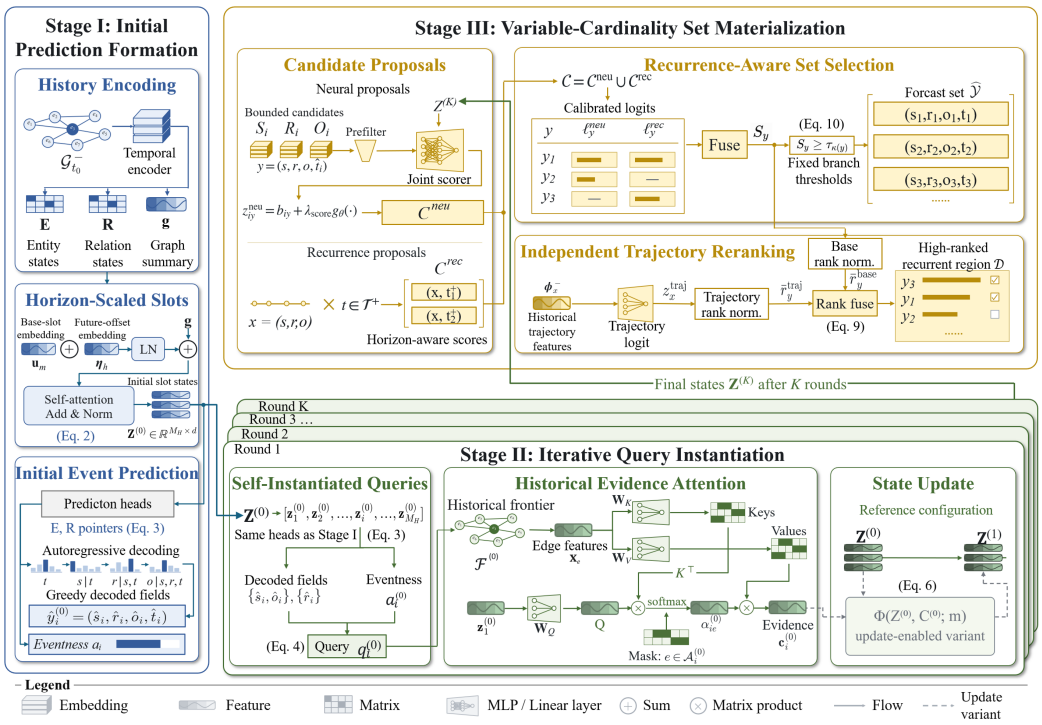



Figure 2: The overall framework of ODCAST.

abstention (Hou et al., 2024). Their link-prediction evaluation supplies some future fields; we distinguish this interface from broader generative capabilities.

**Graph completion and event-sequence forecasting.** Triple Set Prediction completes static graphs without supplied target fields (Zhang et al., 2024). GLEAN predicts event types and candidate actors (Deng et al., 2020); DEF and EventFlow forecast event sequences (Karpukhin & Savchenko, 2026; Kerrigan et al., 2026). OpenForecast and ExAnte benchmark text-based forecasting and ex-ante inference, respectively (Wang et al., 2025; Liu et al., 2026). We instead infer variable-size sets of complete future $(s, r, o, t)$ events, including recurrences, from historical graphs without supplied queries. ODCAST selects evidence through self-instantiated historical queries under a retrieval-edge budget.

## 3 Task Definition

Given a historical TKG $\mathcal{G}_{t_0}^-$ over $(t_0 - W\delta_t, t_0]$, Open-domain forecasting predicts future facts on $\mathcal{T}^+ = \{t_0 + h\delta_t \mid h = 1, \ldots, H\}$, where $t_0$ is the forecast cutoff, $\delta_t > 0$ is the time-step duration, and $W, H$ are the history and forecast lengths in steps. The model outputs an unordered set

$$\widehat{\mathcal{Y}} = f_\theta(\mathcal{G}_{t_0}^-, t_0, H) = \left\{(\hat{s}_j, \hat{r}_j, \hat{o}_j, \hat{t}_j)\right\}_{j=1}^{\widehat{N}}, \tag{1}$$

where $f_\theta$ is the model with parameters $\theta$ and $\widehat{N} = |\widehat{\mathcal{Y}}| \geq 0$ is the inferred output count. Each quadruple specifies a predicted subject, relation, object, and timestamp in $\mathcal{T}^+$. No target fields or counts are supplied. Predictions use entities and relations observed by $t_0$, allowing previously unseen combinations.

# 4 METHOD

As shown in Fig. 2, we propose ODCAST, an open-domain forecasting framework on TKGs, consisting of three stages for initial prediction formation, iterative query instantiation, and variable-cardinality set materialization.

## 4.1 STAGE I: INITIAL PREDICTION FORMATION

A relation-aware temporal encoder maps $\mathcal{G}_{t_0}^-$ to entity states $\mathbf{E}$, relation states $\mathbf{R}$, and graph summary $\mathbf{g}$. With one-step capacity $M_1$, horizon $H$ uses $M_H = HM_1$ slots formed from base vectors $\mathbf{u}_m$, offset embeddings $\boldsymbol{\eta}_h$, and history summary $\mathbf{g}$, all in $\mathbb{R}^d$:

$$\begin{aligned} \bar{\mathbf{z}}_{m,h} &= \mathrm{LN}(\mathbf{u}_m + \boldsymbol{\eta}_h) + \mathbf{g}, \\ \mathbf{Z}^{(0)} &= \mathrm{LN}\big(\bar{\mathbf{Z}} + \mathrm{MHA}(\bar{\mathbf{Z}}, \bar{\mathbf{Z}}, \bar{\mathbf{Z}})\big), \end{aligned} \tag{2}$$

where $\bar{\mathbf{Z}} \in \mathbb{R}^{M_H \times d}$ stacks the history-conditioned slot vectors, LN denotes layer normalization, and MHA performs multi-head self-attention using the same slot matrix as the query, key, and value inputs. For $m = 1, \ldots, M_1$ and $h = 1, \ldots, H$, $i = (h-1)M_1 + m$ indexes flattened slots. $\mathbf{Z}^{(k)} \in \mathbb{R}^{M_H \times d}$ stacks round-$k$ states $\mathbf{z}_i^{(k)}$. Offset embeddings initialize slots without restricting their legal future timestamps.

Let $\Omega^{\mathrm{enc}} \subseteq \Omega$ be the neural decoder's representable domain, determined by the finite-history encoder and legal field masks without future targets. State $\mathbf{z}_i^{(k)}$ predicts non-null probability $a_i^{(k)} \in [0,1]$ and conditional field distributions $p_{i,f}^{(k)}$. For a candidate fact $y = (s, r, o, t)$, $y_f$ and $y_{<f}$ denote field $f$ and preceding fields in order $t, s, r, o$; $\perp$ denotes no emission. Conditioning on the slot state and encoded history is implicit. The slot distribution over $\xi \in \{\perp\} \cup \Omega^{\mathrm{enc}}$ is

$$\widetilde{p}_i^{(k)}(\xi) = \begin{cases} 1 - a_i^{(k)}, & \xi = \perp, \\ a_i^{(k)} \displaystyle\prod_{f \in (t,s,r,o)} p_{i,f}^{(k)}(y_f \mid y_{<f}), & \xi = y \in \Omega^{\mathrm{enc}}. \end{cases} \tag{3}$$

Heads score entity and relation rows of $\mathbf{E}$ and $\mathbf{R}$, with legal identity/time masks. Decoding in order $t, s, r, o$ gives the current hypothesis $\hat{y}_i^{(k)} = (\hat{s}_i^{(k)}, \hat{r}_i^{(k)}, \hat{o}_i^{(k)}, \hat{t}_i^{(k)})$.

## 4.2 STAGE II: ITERATIVE QUERY INSTANTIATION

Self-Instantiated Queries (SIQs) use predicted identities to locate history and uncertainty to allocate access. For $K$ rounds, $k = 0, \ldots, K-1$, let $\bar{p}_{i,E}^{(k)}$, $\bar{p}_{i,R}^{(k)}$, and $p_{i,t}^{(k)}$ be the policy's coarse entity, coarse relation, and time distributions, and $\overline{\mathcal{H}}$ normalized entropy. Priority $u_i^{(k)}$ and query $q_i^{(k)}$ are

$$\begin{aligned} u_i^{(k)} &= 1 - |2a_i^{(k)} - 1| + \overline{\mathcal{H}}(\bar{p}_{i,E}^{(k)}) + \overline{\mathcal{H}}(\bar{p}_{i,R}^{(k)}) + \overline{\mathcal{H}}(p_{i,t}^{(k)}), \\ q_i^{(k)} &= \big(i, \{\hat{s}_i^{(k)}, \hat{o}_i^{(k)}\}, \{\hat{r}_i^{(k)}\}, \ell, b_i^{(k)}\big), \end{aligned} \tag{4}$$

where $\ell$ is the query lookback and $b_i^{(k)}$ is its nonnegative integer access quota. Slots are prioritized by $u_i^{(k)}$; keys are deduplicated with slot attribution retained. Within each window, $B$ caps the number of unique historical edges scored by the retrieval module across all rounds. This cap does not bound history encoding, recurrence preprocessing, or candidate scoring. Queries use predicted rather than target fields.

The temporal index returns historical frontier $\mathcal{F}^{(k)}$ within the lookback, with attributed support $\mathcal{A}_i^{(k)} \subseteq \mathcal{F}^{(k)}$ for slot $i$. For an edge feature $\mathbf{x}_e \in \mathbb{R}^{d_x}$, projections $\mathbf{W}_Q \in \mathbb{R}^{d \times d}$ and $\mathbf{W}_K, \mathbf{W}_V \in \mathbb{R}^{d \times d_x}$ produce queries, keys, and values. For nonempty $\mathcal{A}_i^{(k)}$, attention weights and evidence are

$$\begin{aligned} \alpha_{ie}^{(k)} &= \mathrm{softmax}_{e \in \mathcal{A}_i^{(k)}}\left(\frac{(\mathbf{W}_Q \mathbf{z}_i^{(k)})^\top \mathbf{W}_K \mathbf{x}_e}{\sqrt{d}}\right), \\ \mathbf{c}_i^{(k)} &= \sum_{e \in \mathcal{A}_i^{(k)}} \alpha_{ie}^{(k)} \mathbf{W}_V \mathbf{x}_e, \end{aligned} \tag{5}$$

where $\mathbf{C}^{(k)} \in \mathbb{R}^{M_H \times d}$ stacks the evidence vectors $\mathbf{c}_i^{(k)}$. Empty supports give zero vectors without evaluating the softmax. Training uses soft attention; inference renormalizes weights over top-scoring edges. Non-access costs are recorded separately.

For evidence-based refinement, $\mathbf{m} \in \{0,1\}^{M_H}$ masks eligible evidence. Set $\widetilde{\mathbf{C}} = \mathrm{diag}(\mathbf{m})\mathbf{C}^{(k)}$:

$$\begin{aligned}
\mathbf{U} &= \mathrm{LN}_c\Big(\mathbf{Z}^{(k)} + \mathrm{Drop}\Big(\mathrm{MHA}_c(\mathbf{Z}^{(k)}, \widetilde{\mathbf{C}}, \widetilde{\mathbf{C}}; \mathbf{m})\Big)\Big), \\
\mathbf{V} &= \mathrm{LN}_s(\mathbf{U} + \mathrm{Drop}(\mathrm{MHA}_s(\mathbf{U}, \mathbf{U}, \mathbf{U}))), \\
\mathbf{P} &= \mathrm{FFN}([\mathbf{V} \| \widetilde{\mathbf{C}}]), \\
\mathbf{G} &= \sigma\Big(\mathrm{Linear}_g([\mathbf{Z}^{(k)} \| \mathbf{P} \| \widetilde{\mathbf{C}}])\Big), \\
\mathbf{Z}^{(k+1)} &= \Phi(\mathbf{Z}^{(k)}, \mathbf{C}^{(k)}; \mathbf{m}) \\
&= \begin{cases} \mathbf{Z}^{(k)}, & \sum_i m_i = 0, \\ \mathrm{LN}_o\big(\mathbf{Z}^{(k)} + \mathbf{G} \odot (\mathbf{P} - \mathbf{Z}^{(k)})\big), & \text{otherwise.} \end{cases}
\end{aligned} \tag{6}$$

where $\|$ concatenates features, $\odot$ multiplies elementwise, and $\sigma$ is sigmoid. $\mathbf{U}$ attends across valid evidence summaries; $\mathbf{V}$ mixes slots by self-attention. The $2d \to d \to d$ FFN uses GELU and dropout between affine layers to form proposal $\mathbf{P}$; the gate maps $3d$ features to $d$ gates $\mathbf{G}$. The three LNs have separate parameters; dropout is disabled at evaluation. The mask restricts evidence, not which slots may change.

After re-decoding, the auxiliary increment $\delta_i = m_i \,\mathrm{Linear}_a([\mathbf{z}_i^{(k+1)} \| \widetilde{\mathbf{c}}_i])$ is added only to previously inactive slots' non-null logits, where $\widetilde{\mathbf{c}}_i$ is row $i$ of $\widetilde{\mathbf{C}}$. The identity-update reference omits state refinement, activation increments, and evidence-derived candidate expansion.

### 4.3 Stage III: Variable-Cardinality Set Materialization

**Candidate construction and source fusion.** Final slots supply bounded subject/relation/object lists $\mathcal{S}_i$, $\mathcal{R}_i$, and $\mathcal{O}_i$, expanded using cutoff-safe relation–role statistics. Candidate combinations are prefiltered before joint scoring. For $y = (s, r, o, \hat{t}_i)$ with final timestamp $\hat{t}_i \equiv \hat{t}_i^{(K)}$, the slot–candidate score is

$$\begin{aligned}
z_{iy}^{\mathrm{neu}} &= b_{iy} + \lambda_{\mathrm{score}} g_\theta(\mathbf{z}_i^{(K)}, \mathbf{E}_s, \mathbf{R}_r, \mathbf{E}_o), \\
\mathcal{C} &= \mathcal{C}^{\mathrm{neu}} \cup \mathcal{C}^{\mathrm{rec}} \subseteq \Omega,
\end{aligned} \tag{7}$$

where $b_{iy}$ is the cached field score, $g_\theta$ is a joint multilayer perceptron (MLP) weighted by $\lambda_{\mathrm{score}}$, and $\mathbf{E}_s$, $\mathbf{R}_r$, and $\mathbf{E}_o$ are candidate identity states. These scores select each slot's neural proposal in $\mathcal{C}^{\mathrm{neu}}$. Recurrence proposals $\mathcal{C}^{\mathrm{rec}}$ pair historical triples $x = (s, r, o)$ with legal future timestamps using horizon-aware scores; their union $\mathcal{C}$ is deduplicated by quadruple.

For source $b \in \{\mathrm{rec}, \mathrm{neu}\}$, availability $v_y^b \in \{0,1\}$ and independently calibrated logit $\ell_y^b$ refer to deduplicated candidate $y$, whereas $z_{iy}^{\mathrm{neu}}$ is slot-specific. Label $\kappa(y) \in \{\mathrm{rec}, \mathrm{nov}\}$ is rec when triple $(s, r, o)$ occurs in $\mathcal{G}_{t_0}^-$ and nov otherwise, independently of source availability. The emission score is

$$S_y = \begin{cases} \alpha v_y^{\mathrm{rec}} \ell_y^{\mathrm{rec}} + \beta v_y^{\mathrm{neu}} \ell_y^{\mathrm{neu}} + \gamma v_y^{\mathrm{rec}} v_y^{\mathrm{neu}}, & \kappa(y) = \mathrm{rec}, \\ \ell_y^{\mathrm{neu}}, & \kappa(y) = \mathrm{nov}, \end{cases} \tag{8}$$

where $\alpha$ and $\beta$ weight the recurrence and neural logits, and $\gamma$ is an offset for dual-source candidates. Fusion weights and branch thresholds $\tau_{\mathrm{rec}}, \tau_{\mathrm{nov}}$ are fixed after calibration. Absent-source contributions are zero.

**Independent ranking and set selection.** An independent logistic expert scores triple $x = (s, r, o)$ as $z_x^{\mathrm{traj}} = \mathbf{w}_{\mathrm{traj}}^\top \boldsymbol{\phi}_x^- + b_{\mathrm{traj}}$, using historical trajectory features $\boldsymbol{\phi}_x^-$ available at cutoff $t_0$, weights $\mathbf{w}_{\mathrm{traj}}$, and bias $b_{\mathrm{traj}}$. It receives no neural scores or slot states; candidates $y = (s, r, o, t)$ sharing a triple share its trajectory logit.

Within high-ranked recurrent region $\mathcal{D} \subseteq \mathcal{C}$, $R_y^{\text{base}}$ and $R_y^{\text{traj}}$ are candidate $y$'s base and trajectory ranks (1 is highest), normalized and fused as

$$\begin{aligned} \bar{r}_y^b &= 1 - \frac{R_y^b - 1}{\max(1, |\mathcal{D}| - 1)}, \qquad b \in \{\text{base}, \text{traj}\}, \\ r_y^{\text{fuse}} &= (1-\eta)\bar{r}_y^{\text{base}} + \eta \bar{r}_y^{\text{traj}}, \end{aligned} \tag{9}$$

where $\bar{r}_y^b$ is the normalized rank score and $\eta$ is the fusion weight. Rank fusion reorders the region's original ranking-score positions without changing $S_y$ or set membership. Reranking is skipped when $\mathcal{D}$ is empty. The recurrent- and novel-branch thresholds determine set membership:

$$\widehat{\mathcal{Y}} = \{y \in \mathcal{C} : S_y \geq \tau_{\kappa(y)}\}, \qquad \widehat{N} = |\widehat{\mathcal{Y}}|. \tag{10}$$

Recurrence proposals supplement neural slots, so $M_H$ does not bound total output count.

**Permutation-invariant learning.** Matching covers the $N_s = |\mathcal{Y}^\star \cap \Omega^{\text{enc}}|$ neural-supported targets. In the field-aware cost matrix $\mathbf{D} \in \mathbb{R}^{M_H \times N_s}$, $D_{ij}$ is the cost of assigning slot $i$ to target $j$ in this subset.

$$\begin{aligned} \mathbf{A}^\star &\in \arg\min_{\mathbf{A} \in \mathcal{U}} \langle \mathbf{A}, \mathbf{D} \rangle, \\ \mathcal{U} &= \left\{ \mathbf{A} \in \{0,1\}^{M_H \times N_s} : \mathbf{A}^\top \mathbf{1}_{M_H} = \mathbf{1}_{N_s},\ \mathbf{A}\mathbf{1}_{N_s} \leq \mathbf{1}_{M_H} \right\}, \end{aligned} \tag{11}$$

where $A_{ij} = 1$ assigns slot $i$ to target $j$, $\langle \mathbf{A}, \mathbf{D} \rangle = \sum_{i,j} A_{ij} D_{ij}$, and $\mathbf{1}_n$ is the length-$n$ all-ones vector. Feasibility requires $N_s \leq M_H$. Matched fields receive negative-log-likelihood supervision and all slots non-null supervision, with auxiliary set, duplicate, joint-ranking, and retrieval-utility terms. The trajectory expert uses separate window-balanced binary cross-entropy training.

## 5 Experiments

We conduct extensive experiments to evaluate the effectiveness of our proposed framework. Specifically, we aim to answer the following Research Questions (RQs):

- **RQ1**: How does ODCAST compare with conventional adapted TKG forecasting methods?
- **RQ2**: How do the full architecture and its individual components affect forecasting performance?
- **RQ3**: How do the historical access budget and retrieval rounds affect historical-support recall and access cost?
- **RQ4**: How do different output-selection rules affect the accuracy of predicted fact counts across forecast windows?
- **RQ5**: How sensitive is forecasting performance to the recurrent- and novel-branch thresholds?

### 5.1 Experimental Settings

**Datasets.** We conduct experiments on ICEWS14, ICEWS05–15, and tkgl-polecat (Gastinger et al., 2024a) with forecast horizons $H \in \{1, 3, 7\}$, each spanning $H\delta_t$, where $\delta_t$ denotes the evaluation time-step duration. Both ICEWS datasets use 30 days of history. Dataset statistics and preprocessing details are provided in Appendix A.1.

**Baselines.** To evaluate the effectiveness of ODCAST, we compare it with five representative TKG forecasting methods: CyGNet (Zhu et al., 2021), RE-GCN (Li et al., 2021b), TiRGN (Li et al., 2022a), CognTKE (Chen et al., 2025), and NADEx (Gan et al., 2026). The recent baselines include cognitive path-based reasoning and negative-aware diffusion. All baselines are adapted to complete-event forecasting using a common history-based generator of subject–relation–time prefixes, without supplied future event fields or event counts. The comparison follows the same fixed-cutoff evaluation protocol and retains all recorded target events. Detailed adaptation settings are provided in Appendix A.3.

Table 1: Main results on three temporal knowledge graph datasets. Bold and underlined values indicate the best overall and baseline results, respectively. Models marked with † are conditional forecasting models adapted to Open-Domain Forecasting.

| Method | $H = 1$ | | | $H = 3$ | | | $H = 7$ | | |
|---|---|---|---|---|---|---|---|---|---|
| | Strict F1 ↑ | Soft F1 ↑ | Card MAE ↓ | Strict F1 ↑ | Soft F1 ↑ | Card MAE ↓ | Strict F1 ↑ | Soft F1 ↑ | Card MAE ↓ |
| **ICEWS14** | | | | | | | | | |
| CyGNet† (2021) | 8.93 | 33.19 | 548.62 | 7.10 | 34.67 | 1415.88 | 6.20 | 35.42 | 2680.44 |
| RE-GCN† (2021b) | 4.40 | 24.62 | 775.56 | 3.75 | 25.31 | 1895.42 | 3.22 | 26.08 | 3987.15 |
| TiRGN† (2022a) | 9.88 | 34.52 | 512.40 | 7.65 | 35.11 | 1330.75 | 6.51 | 36.22 | 2541.30 |
| CognTKE† (2025) | 10.10 | 35.77 | 472.35 | 7.93 | 36.02 | 1252.64 | 6.81 | 37.10 | 2420.97 |
| NADEx† (2026) | 10.41 | 36.31 | 445.28 | 8.16 | 36.44 | 1200.50 | 6.97 | 37.89 | 2368.75 |
| **ODCAST (Ours)** | **14.38** | **50.57** | **135.33** | **11.23** | **53.94** | **150.50** | **9.98** | **55.16** | **234.50** |
| **Improvement (%)** | **+38.18** | **+39.29** | **+69.61** | **+37.56** | **+48.01** | **+87.46** | **+43.15** | **+45.55** | **+90.10** |
| **ICEWS05–15** | | | | | | | | | |
| CyGNet† (2021) | 6.10 | 20.97 | 495.18 | 5.40 | 22.14 | 1268.40 | 4.86 | 20.82 | 3245.61 |
| RE-GCN† (2021b) | 2.64 | 6.96 | 1944.60 | 2.29 | 7.81 | 4823.98 | 2.03 | 7.47 | 11758.87 |
| TiRGN† (2022a) | 6.41 | 21.85 | 462.30 | 5.63 | 22.91 | 1204.55 | 5.01 | 21.34 | 3105.72 |
| CognTKE† (2025) | 6.76 | 22.10 | 441.92 | 5.89 | 23.30 | 1158.77 | 5.29 | 21.87 | 2998.43 |
| NADEx† (2026) | 6.97 | 22.63 | 428.61 | 6.05 | 23.85 | 1126.02 | 5.43 | 22.22 | 2927.38 |
| **ODCAST (Ours)** | **14.70** | **47.86** | **65.46** | **12.22** | **49.46** | **129.10** | **10.88** | **53.24** | **114.28** |
| **Improvement (%)** | **+110.93** | **+111.49** | **+84.73** | **+102.03** | **+107.36** | **+88.53** | **+100.28** | **+139.58** | **+96.10** |
| **tkgl-polecat** | | | | | | | | | |
| CyGNet† (2021) | 5.61 | 42.99 | 410.25 | 5.12 | 43.54 | 735.90 | 4.70 | 43.11 | 1490.34 |
| RE-GCN† (2021b) | 4.66 | 35.37 | 1107.64 | 4.22 | 36.08 | 2210.47 | 3.78 | 35.92 | 4485.23 |
| TiRGN† (2022a) | 5.95 | 44.21 | 385.60 | 5.40 | 44.88 | 702.15 | 4.97 | 44.32 | 1422.68 |
| CognTKE† (2025) | 6.19 | 45.93 | 352.14 | 5.70 | 46.61 | 668.72 | 5.08 | 45.99 | 1355.41 |
| NADEx† (2026) | 6.36 | 46.82 | **317.95** | 5.87 | 47.21 | **645.30** | 5.21 | 46.53 | **1310.85** |
| **ODCAST (Ours)** | **7.95** | **50.45** | 342.07 | **7.10** | **51.33** | 685.40 | **6.41** | **52.18** | 1398.26 |
| **Improvement (%)** | **+25.04** | **+7.76** | −7.59 | **+20.95** | **+8.73** | −6.21 | **+22.84** | **+12.13** | −6.67 |

**Evaluation Metrics.** To evaluate Open-Domain Forecasting, we propose an evaluation protocol using the following metrics, with Strict Micro-F1 as the primary measure.

$$\text{StrictF1} = \frac{2T}{P+G}, \qquad \text{SoftF1} = \frac{2U}{P+G}, \quad \text{CardMAE} = \frac{\sum_w \big||P_w| - |G_w|\big|}{|\mathcal{W}|},$$
$$\text{ExactAP} = \frac{1}{G}\sum_{j=1}^{L} \frac{r_j \sum_{a=1}^{j} r_a}{j}, \quad \text{HSR} = \frac{\sum_w c_w}{\sum_w d_w}, \qquad \text{Edges/window} = \frac{\sum_w |A_w|}{|\mathcal{W}|}, \tag{12}$$

where $\mathcal{W}$ contains forecast windows, $P_w, G_w$ are predicted and target sets, and $T, P, G$ aggregate exact matches, predictions, and targets. $U$ sums optimal one-to-one matching credit within windows ($1/4$ per matched field). For Exact AP ($G > 0$), $L$ counts pooled pre-threshold candidates, and $r_j$ indicates an exact match in its window. For historical-support recall (HSR), $c_w, d_w$ count covered and total distinct recurrent target triples. $A_w$ contains unique historical edges scored across all rounds. All targets are retained, with deduplication within windows only. Details are provided in Appendix A.2.

### 5.2 Result 1: Comparison with Baselines

To answer **RQ1**, we compare ODCAST with five adapted TKG forecasting methods on ICEWS14, ICEWS05–15, and tkgl-polecat at $H \in \{1, 3, 7\}$ under the same fixed-cutoff protocol. As shown in Table 1, ODCAST achieves the highest Strict F1 and Soft F1 across all nine settings. At $H = 1$, the reported relative Strict F1 gains over NADEx, the best-performing baseline, are 38.18%, 110.93%, and 25.04%, respectively. ODCAST also achieves the lowest CardMAE on both ICEWS datasets at every horizon, whereas NADEx yields lower count errors on tkgl-polecat.

Although Strict F1 declines with longer horizons, ODCAST maintains its advantage, supporting its effectiveness in forecasting complete future facts without predefined queries. However, the tkgl-polecat results show that higher fact-level F1 does not necessarily imply more accurate output counts, highlighting the need to evaluate both prediction correctness and output size. Additional analyses of forecast horizons and computational efficiency are provided in Appendices A.6 and A.7, respectively.

### 5.3 Result 2: Ablation Study

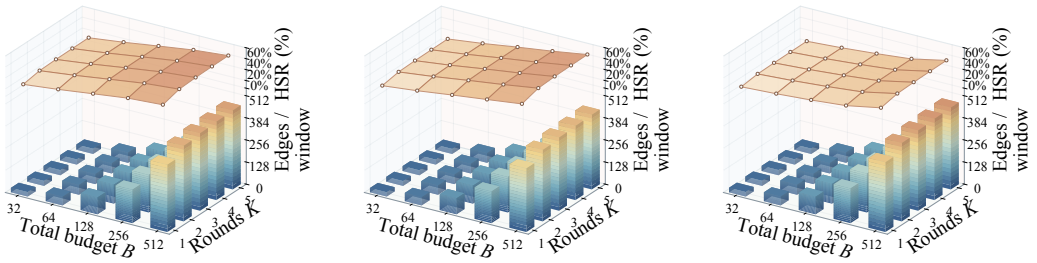
To answer **RQ2**, we conduct ablation studies at $H = 1$ by removing recurrence proposals, recurrence-score fusion, joint tuple scoring, and relation–role expansion, and further compare the full model with the single-pass DirectSet variant. As shown in Table 2, ODCAST achieves the best Strict F1 and Exact AP on all three datasets. Removing recurrence-score fusion causes the largest Strict F1 drops among the component ablations, by 4.32, 4.44, and 1.79 percentage points on ICEWS14, ICEWS05–15, and tkgl-polecat, respectively. DirectSet also performs substantially worse than ODCAST across all datasets. These results indicate that recurrence-aware score fusion plays an important role in forecasting complete future facts, while recurrence proposals, joint tuple scoring, and relation–role expansion also contribute to prediction quality. Detailed component settings and additional analyses are provided in Appendix A.4.

Table 2: Ablation results at $H = 1$. F1 and AP denote Strict F1 and Exact AP, respectively, both reported as percentages. Bold values indicate the best results.

| Configuration | ICEWS14 | | ICEWS05–15 | | tkgl-polecat | |
|---|---|---|---|---|---|---|
| | F1 ↑ | AP ↑ | F1 ↑ | AP ↑ | F1 ↑ | AP ↑ |
| w/o recurrence proposals | 12.76 | 3.14 | 12.96 | 3.64 | 6.86 | 1.43 |
| w/o recurrence-score fusion | 10.06 | 2.24 | 10.26 | 2.43 | 6.16 | 1.13 |
| w/o joint tuple scorer | 12.16 | 3.14 | 12.36 | 3.94 | 6.46 | 2.24 |
| w/o relation–role expansion | 12.56 | 3.44 | 12.86 | 4.14 | 6.76 | 2.38 |
| DirectSet | 6.70 | 1.48 | 7.41 | 1.93 | 3.40 | 0.54 |
| Full ODCAST | **14.38** | **5.25** | **14.70** | **5.92** | **7.95** | **2.50** |

### 5.4 Result 3: Analysis of Access Budget and Retrieval Rounds

To answer **RQ3**, we vary the historical-access budget $B \in \{32, 64, 128, 256, 512\}$ and retrieval rounds $K \in \{1, 2, 3, 4, 5\}$ at $H = 1$, and measure HSR and Edges/window. As shown in Fig. 3, larger budgets consistently improve HSR, while additional rounds provide diminishing gains. At $B = 256$, increasing $K$ from 1 to 2 improves HSR by 5.53–7.84 percentage points across the three datasets, whereas increasing $K$ from 2 to 5 adds only 1.94–2.62 points. Two rounds already retain 92.09–93.91% of the five-round HSR. These results indicate that a moderate number of retrieval rounds captures most of the available historical support, while larger budgets mainly improve coverage at additional access cost. Detailed metric definitions are provided in Appendix A.2.

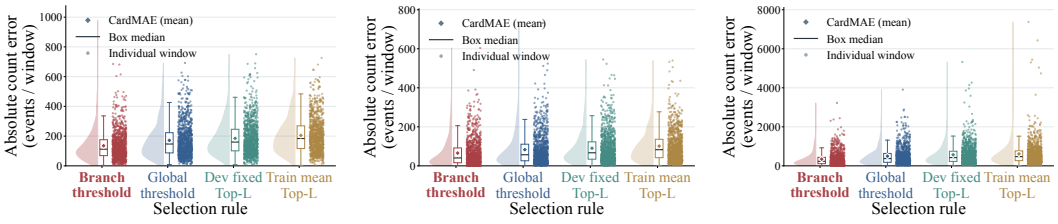



(a) ICEWS14 (b) ICEWS05-15 (c) tkgl-polecat

Figure 3: Effects of historical-access budget $B$ and retrieval rounds $K$ at $H = 1$. Bars show mean unique historical edges scored per window; the surface shows HSR.

### 5.5 Result 4: Output-Cardinality Control Analysis

To answer **RQ4**, we compare branch-specific thresholds with a global threshold, development-selected Top-$L_{\text{emit}}$, and a training-mean count rule at $H = 1$, using the same candidate pools and forecast windows. As shown in Fig. 4, branch-specific thresholds achieve CardMAE values of 135.33, 65.46, and 342.07 on ICEWS14, ICEWS05–15, and tkgl-polecat, reducing count errors by 21.05%, 21.22%, and 31.16% over the global threshold, respectively. These results show that separate selection thresholds provide better control of the predicted set size than a single global threshold. Additional predicted–target count comparisons and selection analyses are provided in Appendix A.5.

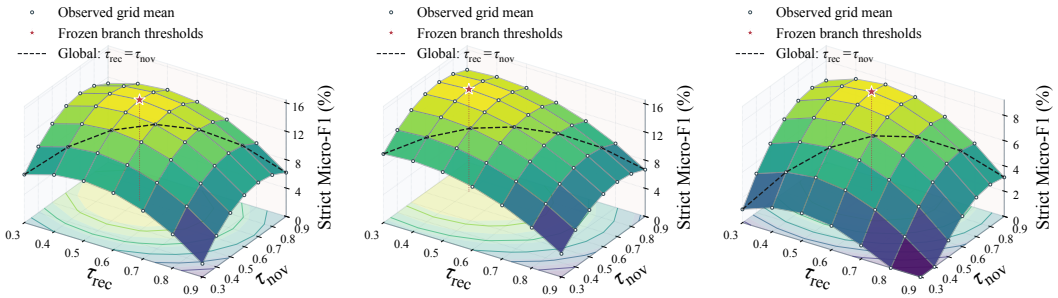



(a) ICEWS14 (b) ICEWS05-15 (c) tkgl-polecat

Figure 4: Absolute output-count errors of four set-selection rules at $H = 1$. Diamonds indicate CardMAE.

### 5.6 Result 5: Branch-Threshold Sensitivity Study

To answer **RQ5**, we vary the recurrent- and novel-branch thresholds $\tau_{\text{rec}}$ and $\tau_{\text{nov}}$ on the development sets at $H = 1$. As shown in Fig. 5, the selected threshold pairs improve Strict F1 over the best global threshold by 0.77, 1.36, and 1.18 percentage points on ICEWS14, ICEWS05–15, and tkgl-polecat, respectively. These results show that separate branch thresholds provide more flexible output selection. Changing either threshold by one grid step (0.1) reduces Strict F1 by at most 0.41, 0.28, and 0.28 points, indicating moderate local sensitivity. Additional parameter analyses are provided in Appendix A.5.




(a) ICEWS14 (b) ICEWS05-15 (c) tkgl-polecat

Figure 5: Strict Micro-F1 sensitivity to recurrent- and novel-event thresholds at $H = 1$. The dashed diagonal denotes a global threshold, and stars mark the selected threshold pairs.

## 6 Conclusion

We formulate and address open-domain forecasting on TKGs, which predicts variable-size sets of complete future relational facts without predefined target queries. To solve this problem, we propose ODCAST, which combines prediction slots, Self-instantiated queries for bounded historical access, and recurrence-aware set materialization with independent trajectory reranking. Experiments on three benchmark datasets demonstrate the effectiveness of ODCAST, while ablation and additional analyses further validate the contributions of its main components and output-selection mechanism.

## REFERENCES

Long Bai, Zixuan Li, Xiaolong Jin, Jiafeng Guo, Xueqi Cheng, and Tat-Seng Chua. G2S: A general-to-specific learning framework for temporal knowledge graph forecasting with large language models. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 20927–20938. Association for Computational Linguistics, 2025.

Yukun Cao, Lisheng Wang, and Luobin Huang. DPCL-Diff: Temporal knowledge graph reasoning based on graph node diffusion model with dual-domain periodic contrastive learning. *Proceedings of the AAAI Conference on Artificial Intelligence*, 39(14):14806–14814, 2025.

Wei Chen, Yuting Wu, Shuhan Wu, Zhiyu Zhang, Mengqi Liao, Youfang Lin, and Huaiyu Wan. CognTKE: A cognitive temporal knowledge extrapolation framework. *Proceedings of the AAAI Conference on Artificial Intelligence*, 39(14):14815–14823, 2025.

Songgaojun Deng, Huzefa Rangwala, and Yue Ning. Dynamic knowledge graph based multi-event forecasting. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 1585–1595. Association for Computing Machinery, 2020.

Yanglei Gan, Peng He, Yuxiang Cai, Run Lin, Guanyu Zhou, and Qiao Liu. Negative-aware diffusion process for temporal knowledge graph extrapolation. In *Findings of the Association for Computational Linguistics: EACL 2026*, pp. 3352–3367, Rabat, Morocco, March 2026. Association for Computational Linguistics.

Julia Gastinger, Shenyang Huang, Mikhail Galkin, Erfan Loghmani, Ali Parviz, Farimah Poursafaei, Jacob Danovitch, Emanuele Rossi, Ioannis Koutis, Heiner Stuckenschmidt, Reihaneh Rabbany, and Guillaume Rabusseau. TGB 2.0: A benchmark for learning on temporal knowledge graphs and heterogeneous graphs. In *Advances in Neural Information Processing Systems*, volume 37, pp. 140199–140229. Curran Associates, Inc., 2024a.

Julia Gastinger, Christian Meilicke, Federico Errica, Timo Sztyler, Anett Schülke, and Heiner Stuckenschmidt. History repeats itself: A baseline for temporal knowledge graph forecasting. In *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence*, pp. 4016–4024. International Joint Conferences on Artificial Intelligence Organization, 2024b.

Zhen Han, Peng Chen, Yunpu Ma, and Volker Tresp. Explainable subgraph reasoning for forecasting on temporal knowledge graphs. In *International Conference on Learning Representations*, 2021.

Zhongni Hou, Xiaolong Jin, Zixuan Li, Long Bai, Jiafeng Guo, and Xueqi Cheng. Selective temporal knowledge graph reasoning. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pp. 14555–14566. ELRA and ICCL, 2024.

Woojeong Jin, Meng Qu, Xisen Jin, and Xiang Ren. Recurrent event network: Autoregressive structure inference over temporal knowledge graphs. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 6669–6683. Association for Computational Linguistics, 2020.

Ivan Karpukhin and Andrey Savchenko. Detecting the future: All-at-once event sequence forecasting with horizon matching. *Proceedings of the AAAI Conference on Artificial Intelligence*, 40(27): 22536–22544, 2026.

Gavin Kerrigan, Kai Nelson, and Padhraic Smyth. EventFlow: Forecasting temporal point processes with flow matching. In *Proceedings of The 29th International Conference on Artificial Intelligence and Statistics*, volume 300 of *Proceedings of Machine Learning Research*, pp. 3124–3132. PMLR, 2026.

Yujia Li, Shiliang Sun, and Jing Zhao. TiRGN: Time-guided recurrent graph network with local-global historical patterns for temporal knowledge graph reasoning. In *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence*, pp. 2152–2158. International Joint Conferences on Artificial Intelligence Organization, 2022a.

Zixuan Li, Xiaolong Jin, Saiping Guan, Wei Li, Jiafeng Guo, Yuanzhuo Wang, and Xueqi Cheng. Search from history and reason for future: Two-stage reasoning on temporal knowledge graphs. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pp. 4732–4743. Association for Computational Linguistics, 2021a.

Zixuan Li, Xiaolong Jin, Wei Li, Saiping Guan, Jiafeng Guo, Huawei Shen, Yuanzhuo Wang, and Xueqi Cheng. 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. Association for Computing Machinery, 2021b.

Zixuan Li, Zhongni Hou, Saiping Guan, Xiaolong Jin, Weihua Peng, Long Bai, Yajuan Lyu, Wei Li, Jiafeng Guo, and Xueqi Cheng. HiSMatch: Historical structure matching based temporal knowledge graph reasoning. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pp. 7328–7338. Association for Computational Linguistics, 2022b.

Yachuan Liu, Xiaochun Wei, Lin Shi, Xinnuo Li, Bohan Zhang, Paramveer Dhillon, and Qiaozhu Mei. ExAnte: A benchmark for ex-ante inference in large language models. In *Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1551–1571. Association for Computational Linguistics, 2026.

Yushan Liu, Yunpu Ma, Marcel Hildebrandt, Mitchell Joblin, and Volker Tresp. TLogic: Temporal logical rules for explainable link forecasting on temporal knowledge graphs. *Proceedings of the AAAI Conference on Artificial Intelligence*, 36(4):4120–4127, 2022.

Rakshit Trivedi, Hanjun Dai, Yichen Wang, and Le Song. Know-Evolve: Deep temporal reasoning for dynamic knowledge graphs. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 3462–3471. PMLR, 2017.

Zhen Wang, Xi Zhou, Yating Yang, Bo Ma, Lei Wang, Rui Dong, and Azmat Anwar. OpenForecast: A large-scale open-ended event forecasting dataset. In *Proceedings of the 31st International Conference on Computational Linguistics*, pp. 5273–5294. Association for Computational Linguistics, 2025.

Gongzhu Yin, Hongli Zhang, Yi Luo, Yuchen Yang, Kun Lu, and Chao Meng. Ignite forecasting with SPARK: An efficient generative framework for refining LLMs in temporal knowledge graph forecasting. In *Database Systems for Advanced Applications (DASFAA 2025)*, volume 15987 of *Lecture Notes in Computer Science*, pp. 746–756. Springer Nature Singapore, 2026.

Wen Zhang, Yajing Xu, Peng Ye, Zhiwei Huang, Zezhong Xu, Jiaoyan Chen, Jeff Z. Pan, and Huajun Chen. Start from zero: Triple set prediction for automatic knowledge graph completion. *IEEE Transactions on Knowledge and Data Engineering*, 36(11):7087–7101, 2024.

Cunchao Zhu, Muhao Chen, Changjun Fan, Guangquan Cheng, and Yan Zhang. Learning from history: Modeling temporal knowledge graphs with sequential copy-generation networks. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(5):4732–4740, 2021.

# A Additional Experimental Details and Analyses

## A.1 Data Construction and Evaluation Manifest

The evaluation covers ICEWS14, ICEWS05–15, and tkgl-polecat. Table 3 retains the dataset and preprocessing fields needed to reproduce the forecast windows. The reported setup uses 30 days of history for all three datasets.

Horizons $H \in \{1, 3, 7\}$ count native evaluation steps and span $H\delta_t$. A horizon is measured in days only when $\delta_t$ is one day. Timestamp conversion, duplicate removal, chronological splits, and identity availability are fixed before evaluation.

Table 3: Dataset and evaluation specifications. Window counts are listed for the train, validation-selection, validation-calibration, and test splits, respectively.

| **Item** | **ICEWS14** | **ICEWS05–15** | **tkgl-polecat** |
|---|---|---|---|
| Time span | 2014-01-01–2014-12-31 | 2005-01-01–2015-12-31 | 2018-01-01–2022-12-31 |
| Entities | 7,128 | 10,488 | 150,931 |
| Relations | 230 | 251 | 16 |
| Time step $\delta_t$ | 1 day | 1 day | 1 day |
| History duration $W\delta_t$ | 30 days | 30 days | 30 days |
| Origin stride | 1 day | 1 day | 1 day |
| *Chronological split spans* | | | |
| Train | 2014-01-01–2014-10-19 | 2005-01-01–2013-10-18 | 2018-01-01–2021-04-07 |
| Validation-selection | 2014-10-20–2014-11-06 | 2013-10-19–2014-05-06 | 2021-04-08–2021-09-18 |
| Validation-calibration | 2014-11-07–2014-11-24 | 2014-05-07–2014-11-23 | 2021-09-19–2022-03-02 |
| Test | 2014-11-25–2014-12-31 | 2014-11-24–2015-12-31 | 2022-03-03–2022-12-31 |
| *Forecast windows (train/select/calibrate/test)* | | | |
| $H = 1$ | 292 / 18 / 18 / 37 | 3,213 / 200 / 201 / 403 | 1,193 / 164 / 165 / 304 |
| $H = 3$ | 290 / 16 / 16 / 35 | 3,211 / 198 / 199 / 401 | 1,191 / 162 / 163 / 302 |
| $H = 7$ | 286 / 12 / 12 / 31 | 3,207 / 194 / 195 / 397 | 1,187 / 158 / 159 / 298 |
| *Deduplicated event counts* | | | |
| All | 90,730 | 461,329 | 1,779,610 |
| Train | 71,432 | 366,174 | 1,246,556 |
| Validation | 10,382 | 45,025 | 266,736 |
| Test | 8,916 | 50,130 | 266,318 |

Each future target grid lies entirely within its chronological split. Historical context may precede the split boundary but contains only observations available at the forecast cutoff. Under the rolling-origin protocol, earlier test observations become available at later origins. Observations within the current forecast horizon are never revealed while forecasting that window. Model parameters and calibration remain frozen throughout test evaluation.

All methods follow the same cutoff and history-window restrictions. Retrieval, recurrence, and trajectory features use the same permitted history $\mathcal{G}_{t_0}^{-}$. An additional historical cache outside this window is not part of the current input specification. Cutoff-visible identity counts, neural support rates, and empty-target rates are derived from this specification and recorded with the processed data. Completed training seeds are reported with the statistical results.

## A.2 Metric Definitions and Statistical Reporting

Let $\mathcal{W}$ denote the evaluated forecast windows. For window $w$, $G_w$, $P_w$, and $C_w$ are the full recorded target set, emitted prediction set, and candidate set before thresholding. Predictions and candidates are deduplicated by complete quadruple within each window. Predictions of the same event from different origins belong to distinct forecast tasks and are not deduplicated across windows.

**Strict event-set accuracy.** Define the aggregate counts

$$T = \sum_{w \in \mathcal{W}} |P_w \cap G_w|, \qquad P = \sum_{w \in \mathcal{W}} |P_w|, \qquad G = \sum_{w \in \mathcal{W}} |G_w|.$$

Strict precision, recall, and Micro-F1 are $T/P$, $T/G$, and $2T/(P+G)$ when their denominators are positive. The implementation assigns precision 1 to an empty prediction set and recall 1 to an empty gold set. F1 is 1 when both aggregate sets are empty and 0 when exactly one is empty. Micro-F1 is computed from aggregate counts rather than averaging window F1. All recorded targets, including unsupported identities, remain in $G_w$.

Empty-set accuracy measures agreement between predicted and target empty/nonempty status over all evaluated windows:

$$\mathrm{EmptyAcc} = \frac{1}{|\mathcal{W}|} \sum_{w \in \mathcal{W}} \mathbb{1}\big[(|P_w| = 0) = (|G_w| = 0)\big].$$

This statistic includes windows in which both sets are nonempty.

**Soft event-set accuracy.** For predicted event $p$ and target event $g$, define role-sensitive field agreement

$$a(p, g) = \frac{1}{4} \sum_{f \in \{s,r,o,t\}} \mathbb{1}[p_f = g_f].$$

Let $\mathfrak{M}_w$ contain the one-to-one matchings between $P_w$ and $G_w$. With unmatched events receiving zero credit,

$$U_w = \max_{M \in \mathfrak{M}_w} \sum_{(p,g) \in M} a(p, g), \qquad F_{1,\mathrm{soft}} = \frac{2 \sum_w U_w}{P + G}.$$

The matching weights are nonnegative, and time receives credit only for an exact timestamp match. Strict and Soft F1 use identical emitted sets and full targets. Soft F1 diagnoses partial agreement rather than replacing exact-event accuracy. It uses the same empty-set conventions as Strict F1.

**Cardinality and candidate coverage.** We report

$$\mathrm{CardMAE} = \frac{1}{|\mathcal{W}|} \sum_{w} \big||P_w| - |G_w|\big|, \qquad \mathrm{Coverage} = \frac{\sum_w |C_w \cap G_w|}{G}.$$

Coverage measures complete future quadruples. Count bias is the mean of $|P_w| - |G_w|$, and normalized absolute count error is $\sum_w ||P_w| - |G_w||/G$. These statistics distinguish the direction of count error from changes in event volume across horizons.

**Exact event average precision.** Exact AP pools candidate predictions across the evaluated windows of one dataset, forecast horizon, and training run. "Exact" specifies complete-quadruple matching, while pooling specifies the aggregation. A candidate is identified by its window and complete quadruple $(w, y)$. Each $C_w$ is deduplicated before pooling, while predictions from different windows remain distinct. Candidates below the emission threshold are included. Let $((w_1, y_1), \ldots, (w_L, y_L))$ denote the combined ranking by the final ranking score, and let $r_j = \mathbb{1}[y_j \in G_{w_j}]$. For total gold count $G > 0$,

$$\mathrm{AP}_{\mathrm{exact}} = \frac{1}{G} \sum_{j=1}^{L} r_j \frac{\sum_{a=1}^{j} r_a}{j}.$$

All gold events remain in the denominator, including events absent from the candidate pool. An empty candidate list receives zero AP when $G > 0$. The metric is computed separately for each run before aggregation across training seeds. It is not the mean of window-level AP values. Equal scores retain the original candidate-list order through the sorting key $(-\mathrm{score}, \mathrm{index})$.

The above Exact AP definition applies to evaluations with $G > 0$.

Candidate-list truncation is reported explicitly. Trajectory reranking changes ranking-score positions within its designated region and preserves emission scores. Exact AP is computed from the resulting pooled ranking.

**Historical support diagnostic.** Let $T_w^{\text{rec}}$ be the distinct triples underlying gold events that also occur in the allowed history. Let $A_w$ contain the unique historical edges scored across retrieval rounds, and let $X(A_w)$ extract their triples. Historical-support recall (HSR) is

$$R_{\text{support}} = \frac{\sum_w |X(A_w) \cap T_w^{\text{rec}}|}{\sum_w |T_w^{\text{rec}}|}.$$

HSR measures historical triple-support coverage, not future-event accuracy. It is computed retrospectively from scored edge identities; gold is not used to construct queries.

**Frozen selection and uncertainty.** Checkpoint selection and calibration use chronological development data. The development partitions, seed list, candidate rules, thresholds, fusion weights, and reranking-region rule are fixed before test evaluation. For analyses with repeated runs, per-run metrics are summarized by their mean and sample standard deviation. Paired differences use identical seeds and forecast origins.

The figures use identifiers 11, 22, and 33. The RQ4 and count-correspondence distributions use identifier 11. These identifiers are labels and do not by themselves establish completed independent training runs.

The proposed uncertainty analysis uses paired moving-block bootstrap resampling over ordered origins. Each replicate recomputes the metric from its aggregate counts or the pooled candidate ranking. The replicate count, seed resampling hierarchy, and block duration are specified before test results are examined. Block duration is selected on development data to account for both overlapping horizons and dependence induced by shared history.

The longest forecast horizon provides a lower bound on block duration but does not guarantee independent blocks. Origin stride determines the corresponding number of windows. Sensitivity to longer blocks, the effective number of blocks, and the completed seed count accompany the intervals. Seed standard deviations and temporal-block intervals are reported separately.

### A.3 Baseline Adaptation Details

All conditional baselines receive the same history-generated subject–relation–time prefixes and prefix limits. Proposed times lie on $\mathcal{T}^+$, and local and global historical inputs are restricted to $\mathcal{G}_{t_0}^-$. Gold prefixes, gold counts, future-active identity lists, and within-horizon observations are excluded. Calibration and threshold selection use only development data and are frozen before testing. Full-target evaluation assesses each scorer together with its adapter. Candidate limits and realized computational costs are recorded. Optional external static-graph inputs are disabled.

For NADEx, the adapter disables the auxiliary entity–word static graph and replaces gold-query preprocessing with autonomous prefixes. Subject histories and derived tensors are rebuilt from the permitted window. Target-time intervals are checked for each proposed future timestamp, and diffusion seeds and sampling budgets are fixed. TiRGN, CognTKE, and NADEx follow the same cutoff-valid prefix, candidate-limit, calibration, and full-target evaluation requirements as the other adapted baselines reported in Table 1.

### A.4 Additional Results and Component Comparisons

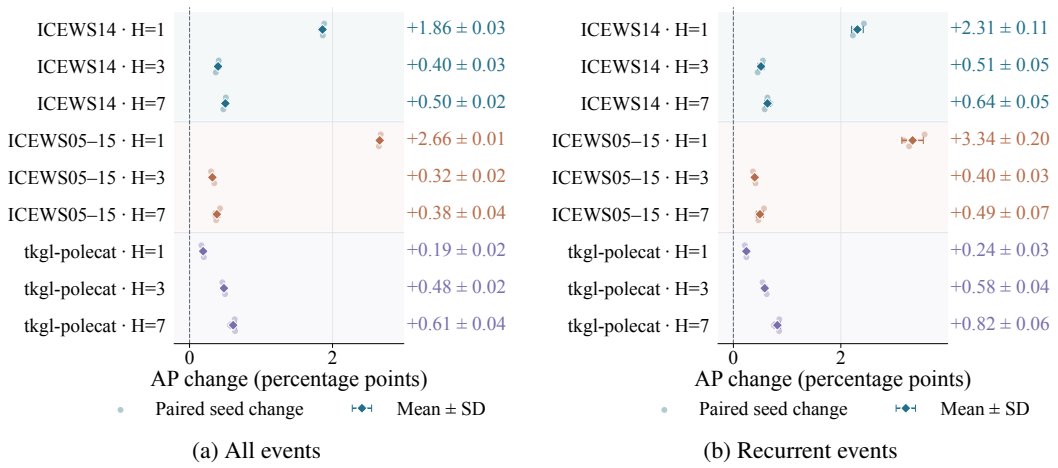
Table 2 reports Strict F1 and Exact AP for the component-reduced variants and DirectSet at $H = 1$. Development measurements retain their split labels and are distinct from test results. The populated ablation table in the main text retains Exact AP to examine trajectory ranking alongside Strict F1.

**DirectSet configuration.** DirectSet provides a single-pass slot-decoding comparison without the additional components listed in RQ2. DirectSet is a joint architectural comparison; matching neural slot counts alone does not ensure equal total output capacity.

The component comparisons describe differences among the evaluated variants in emitted-set accuracy and candidate ranking. With candidates, emission scores, and thresholds fixed, trajectory reranking changes candidate order but not the emitted sets.

Figure 6 displays reranked AP minus pre-reranking AP for all events and recurrent events. Points show paired seed differences, and diamonds and error bars show their means and sample standard

deviations. The zero line denotes unchanged AP. Positive differences indicate higher AP after reranking.




(a) All events (b) Recurrent events

Figure 6: Effects of trajectory reranking on Exact AP. Rows correspond to dataset–horizon pairs.

### A.5 Selection and Hyperparameter Sensitivity

Selection controls compare calibrated branch thresholds, a calibrated global threshold, development-selected fixed Top-$L_{\text{emit}}$, and a training-mean count rule. All four rules use the same candidate pools and emission scores $S_y$. Here $L_{\text{emit}}$ denotes output-list size and is distinct from retrieval rounds $K$.

The training-mean rule rounds the mean full-target count for the relevant horizon to the nearest nonnegative integer. It emits at most the available candidate count, and empty predictions remain valid. The development scan in Figure 5 varies $(\tau_{\text{rec}}, \tau_{\text{nov}})$ and marks the selected branch pair and global-threshold diagonal. CardMAE accompanies Strict F1 in the numerical selection record.

Figure 7 compares predicted and observed counts under branch thresholds and fixed Top-$L_{\text{emit}}$ on the same windows. Hexagons and marginal histograms count windows for the same selected seed, and the dashed line denotes $y = x$. Both rules share coordinate limits, bins, and hexagon colors within a dataset. Thresholds act on emission scores $S_y$, which need not be probabilities.

The displayed development scans vary $M_1$, $K$, $\lambda_{\text{score}}$, and $\eta$. Final settings are frozen before test evaluation. Responses are Strict F1 for $M_1$ and $\lambda_{\text{score}}$, HSR for $K$, and Exact AP for $\eta$. Activity-bin boundaries are fixed using training data before retrospective stratification. Empty-target windows remain separate, and each bin's F1 is recomputed from aggregate counts.

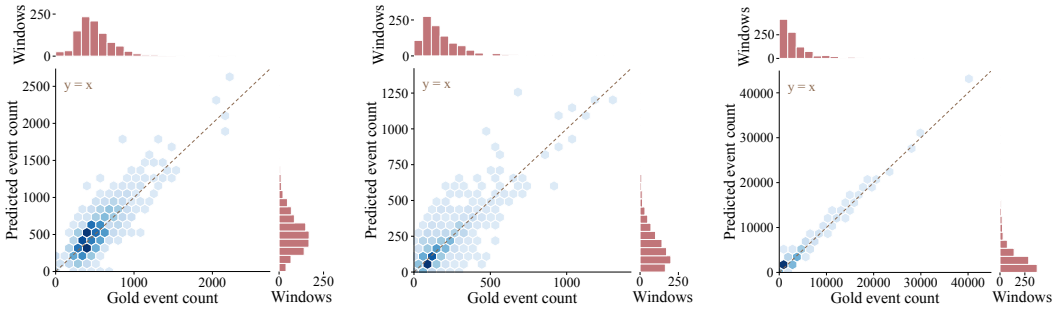
Figure 8 shows the $M_1$, $K$, $\lambda_{\text{score}}$, and $\eta$ scans with other settings fixed. The reference $K$ scan measures HSR, while the $\eta$ scan measures Exact AP. Thin lines show individual seeds, and solid lines and bands show means and sample standard deviations. Markers identify defaults. Lines connect evaluated settings.

History-length, candidate-count, and budget studies require separate cost records. Zero-access conditions are shown separately from logarithmic budget axes. Trajectory-weight studies retain the emission-invariance check. The four displayed scans do not establish these additional controls.

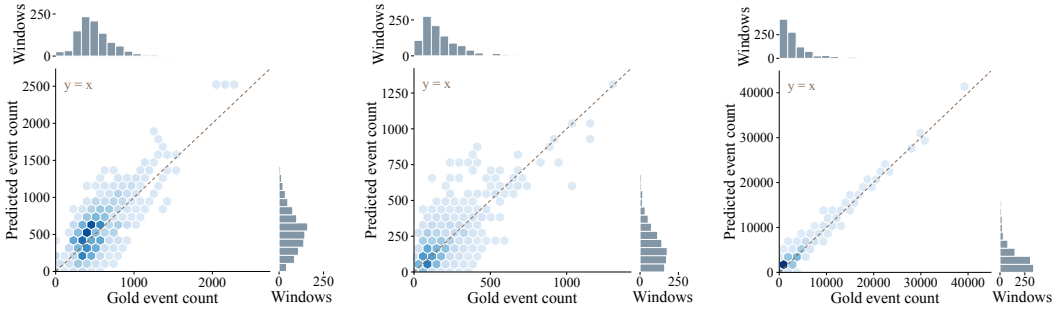
### A.6 Recurrence, Support, and Robustness Slices

Events are recurrent when their exact triple appears in the permitted history and novel otherwise. Figure 9 presents Strict F1 and Exact AP for these groups at $H = 1$. Figure 10 compares both metrics across $H \in \{1, 3, 7\}$ for ODCAST and the comparison methods.

The planned horizon comparison scales neural capacity as $M_H = HM_1$ and holds the retrieval budget fixed. Horizon-scaled budgets form a separate control. These experimental settings require




(a) ICEWS14 (b) ICEWS05-15 (c) tkgl-polecat




(d) ICEWS14 (e) ICEWS05-15 (f) tkgl-polecat

Figure 7: Predicted versus observed event counts at $H = 1$. The top and bottom rows use branch thresholds and fixed Top-$L_{\text{emit}}$, respectively.

verification against the corresponding configuration and run records. Recurrent and novel recall, candidate coverage, and gold counts accompany the corresponding numerical analysis.

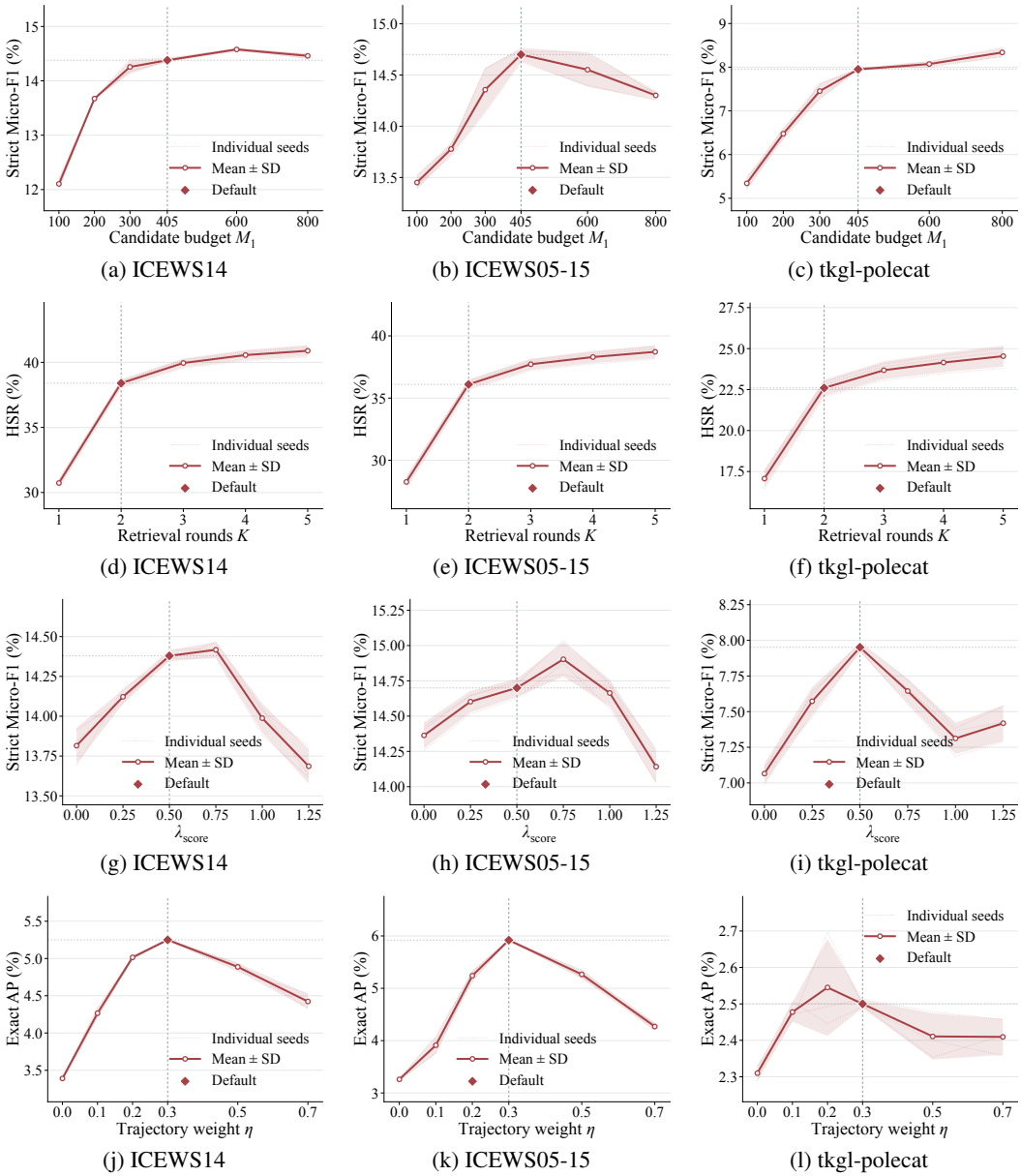
Support-based analysis distinguishes neural-representable events, admissible events outside neural support, and events outside the cutoff-visible vocabulary. Novel triples do not imply unseen identities. The event-type matrices retain zero values and may use different color ranges. Bold entries identify the largest means without implying statistical significance. Horizon curves connect sampled native steps.

**Coverage by candidate source.** For each window, the full gold set is partitioned into events covered only by $C_w^{\text{neu}}$, only by $C_w^{\text{rec}}$, by both, or by neither. Summing these disjoint categories across windows gives the complete coverage decomposition in Figure 11 at $H = 1$. The four shares sum to 100% because their denominator is the full gold set.

Membership dots identify the candidate sources. Numerical neural, recurrence, and union coverage accompany the diagram. This analysis measures gold-event coverage rather than overlap among all proposed candidates. Recurrence proposals contain historically observed triples; novel triples can enter the candidate union only through the neural source.

**Conditional timing error.** The supplementary timing diagnostic pairs predictions and targets sharing an exact triple. It uses maximum-cardinality one-to-one timestamp matching and minimizes absolute time differences among such matchings. The resulting mean absolute error is measured in units of $\delta_t$. The paired fraction of all gold events and overall Strict F1 accompany it, because unmatched events receive no timing-error value.

**Historical triple-frequency analysis.** Figure 12 groups test gold events by the historical occurrence count of their exact triple, using bins Unseen, 1, 2–5, 6–20, and $> 20$. Upper bars give the test gold-event count in each historical-frequency bin; these counts are shared across methods and seeds. Lower matrices show complete-quadruple recall by method, summarized by seed means and sample standard deviations. Unseen denotes a triple absent from permitted history, rather than an unseen entity or relation.



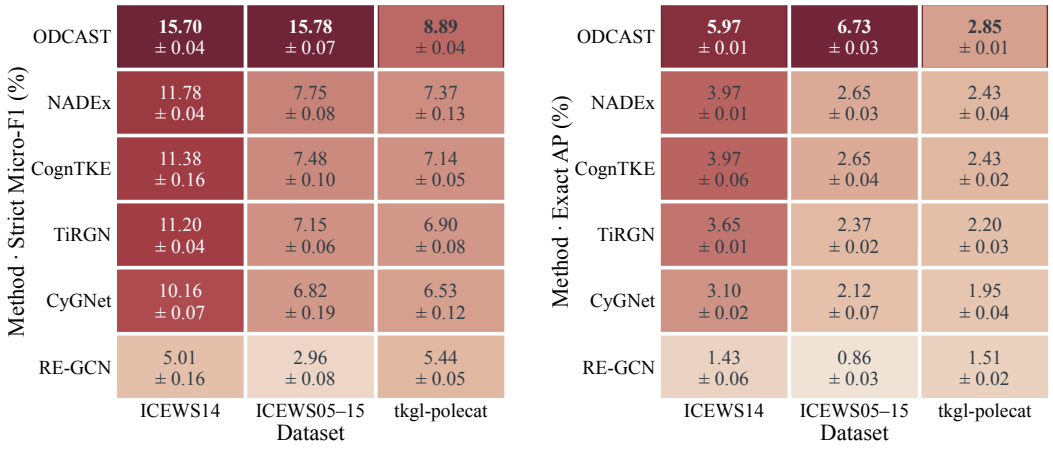

(a) ICEWS14 (b) ICEWS05-15 (c) tkgl-polecat

(d) ICEWS14 (e) ICEWS05-15 (f) tkgl-polecat

(g) ICEWS14 (h) ICEWS05-15 (i) tkgl-polecat

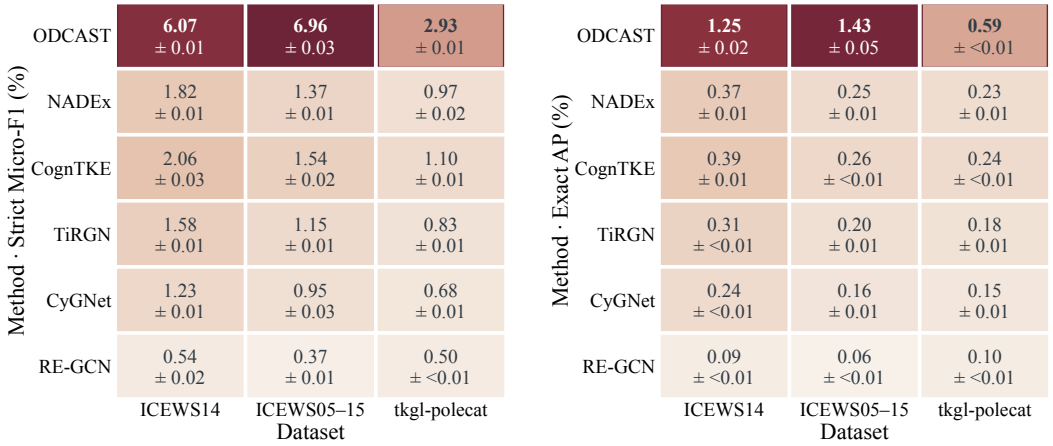
(j) ICEWS14 (k) ICEWS05-15 (l) tkgl-polecat

Figure 8: Parameter sensitivity analysis. Rows vary $M_1$, $K$, $\lambda_{\text{score}}$, and $\eta$ from top to bottom, with Strict F1, HSR, Strict F1, and Exact AP as the respective metrics. Columns show ICEWS14, ICEWS05–15, and tkgl-polecat.




(a) Recurrent events (Strict F1)

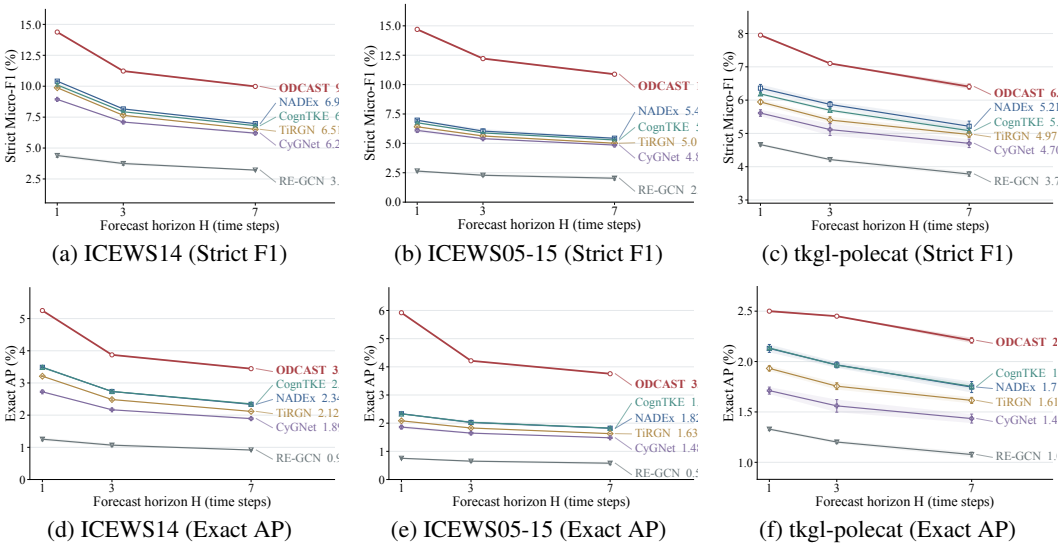
(b) Recurrent events (Exact AP)




(c) Novel events (Strict F1)

(d) Novel events (Exact AP)

Figure 9: Forecasting results for recurrent and novel events at $H = 1$. Values are means $\pm$ standard deviations (%).




(a) ICEWS14 (Strict F1)

(b) ICEWS05-15 (Strict F1)

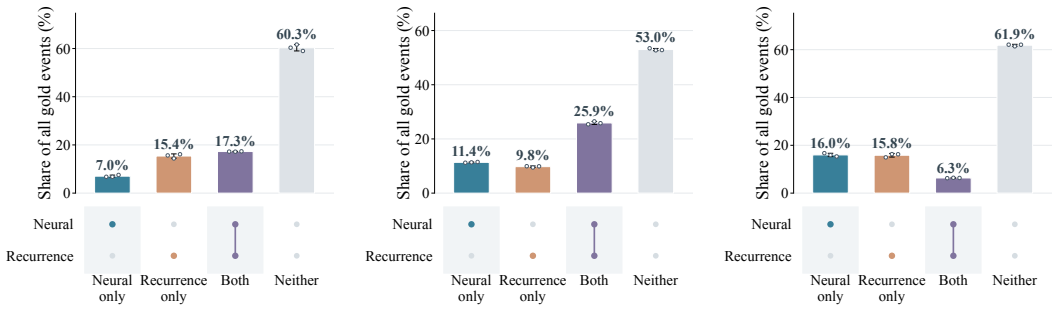
(c) tkgl-polecat (Strict F1)

(d) ICEWS14 (Exact AP)

(e) ICEWS05-15 (Exact AP)

(f) tkgl-polecat (Exact AP)

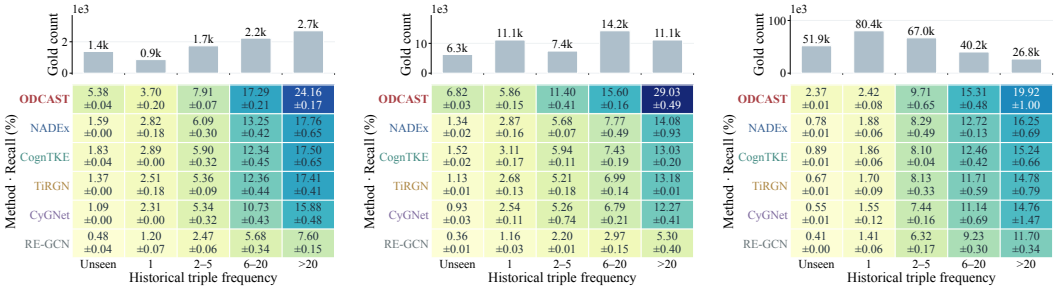
Figure 10: Forecasting performance across horizons $H \in \{1, 3, 7\}$. Lines and shaded bands indicate means and standard deviations.




(a) ICEWS14 (b) ICEWS05-15 (c) tkgl-polecat

Figure 11: Gold-event coverage by candidate source at $H = 1$. Categories represent coverage by neural candidates only, recurrence candidates only, both sources, or neither. Error bars indicate standard deviations.

Additional relation-frequency and historical-gap analyses use boundaries selected on training or development data and retain sparse bins. They are distinct from the plotted triple-frequency analysis. History-removal controls apply the same cutoff-valid masking rule to all methods and distinguish random removal from removal of recent observations. All grouping occurs after prediction and supplies no target-dependent routing information to the model.




(a) ICEWS14 (b) ICEWS05-15 (c) tkgl-polecat

Figure 12: Historical triple-frequency analysis at $H = 1$. The top panels show the number of test gold events in each historical-frequency bin, and the bottom panels report quadruple recall as means $\pm$ standard deviations across seeds.

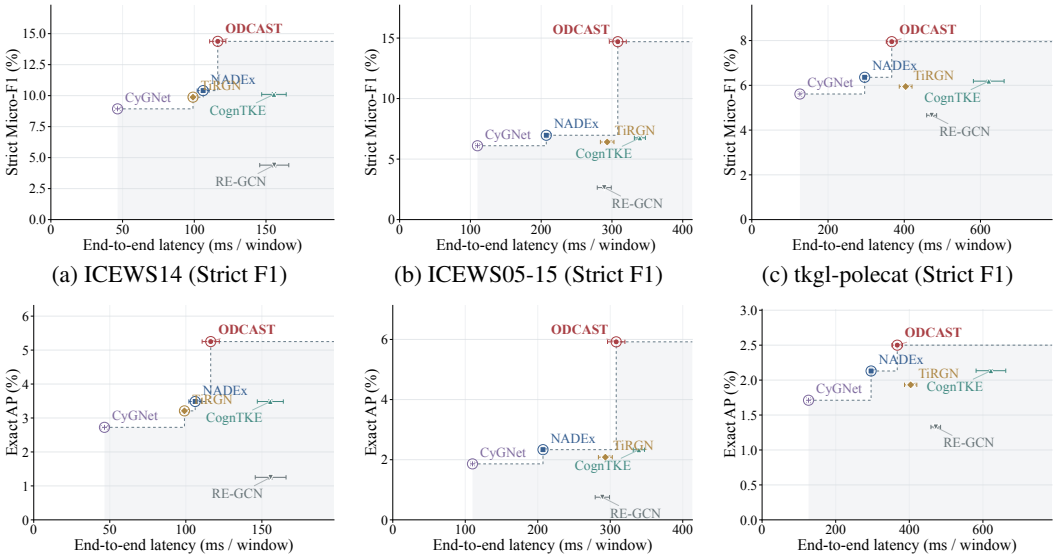
### A.7 Efficiency Comparison

Figure 13 illustrates quality versus latency at $H = 1$, with Strict F1 and Exact AP in separate rows. Non-dominated points are identified among the plotted configurations for each dataset and metric. Shading marks the region dominated by these evaluated points.

The measurement record specifies hardware, software versions, numerical precision, batch size, cache policy, warm-up, synchronization, and timed window count. Figure 13 reports mean latency in ms/window against Strict F1 and Exact AP. Component timings separate history encoding, recurrence preprocessing, retrieval, candidate scoring, and selection or reranking. Shared and amortized computation is identified explicitly.

Scaling analyses compare fixed and horizon-scaled budgets and vary history graph size through reproducible, target-independent subsampling while retaining the original future targets. A full-history endpoint anchors this controlled study. It is distinct from grouping naturally different windows by observed history size.

Actual node, edge, and candidate counts accompany timing measurements, since a retrieval cap does not bound total cost. Cold-start and steady-state cache measurements are reported separately, with precomputation amortization and the number of timed windows specified.




(a) ICEWS14 (Strict F1) (b) ICEWS05-15 (Strict F1) (c) tkgl-polecat (Strict F1)

(d) ICEWS14 (Exact AP) (e) ICEWS05-15 (Exact AP) (f) tkgl-polecat (Exact AP)

Figure 13: Forecasting quality versus inference latency at $H = 1$. Latency is measured in milliseconds per window. Error bars indicate standard deviations, and rings mark Pareto-optimal points among the compared configurations.