When Do Answers Change? Estimating Question Recency Demands in QA with Multi-Events

Peter Schulze
github.com/peterschulze04
Peter.Schulze@student.uibk.ac.at

Fabian Stiewe
github.com/Fabianstw
Fabian.Stiewe@student.uibk.ac.at

Abstract

Large language models (LLMs) answer questions using knowledge that may change over time; however, most evaluations remain static. This work explores the demands of question recency, specifically the frequency with which answers must be updated to maintain accuracy, and the precision with which models can predict the necessary update frequency. We construct a *generation-and-labeling* pipeline that produces recency-aware questions along two axes: stationary vs.versus non-stationary temporal behaviour, and single-event vs. multi-event dependencies. The multi-event category captures questions whose answers rely on multiple temporally connected events, including causal and purely co-occurring pairs. The single-event category focuses on questions about a single evolving process. Stationarity labels distinguish predictable update cycles from volatile changes. Employing a Llama model, the pipeline generates and labels a $\sim$1400-question dataset, assigning recency classes, stationarity labels, and multi-event markers. Subsequently, it applies perturbations to construct more challenging temporal variants. The resulting dataset extends prior work, providing new and more demanding questions with which to evaluate LLMs on when those answers change.

1 Introduction

In the last couple of years, large language models (LLMs) have demonstrated remarkable capabilities across a wide range of natural language processing tasks. In particular, their performance on question answering (QA) tasks has been studied extensively (Fischer et al., 2024; Brown et al., 2020), with very strong results. However, a persistent challenge for QA systems is handling temporal dynamics, specifically answering questions whose correct answers change over time.

Much of this prior work largely ignores the fact that many real-world questions require information that is time-sensitive and may change frequently, such as:

- *"What is the current population of Germany?"*
- *"Who is the current president of the United States?"*

As facts evolve, answers that were once correct inevitably degrade, and outdated responses can harm user trust. Despite this, many QA evaluations remain static, implicitly assuming that factual correctness is invariant over time.

Prior work on temporal QA and time-aware modeling has highlighted this limitation from several perspectives. These include datasets for time-sensitive questions (Chen et al., 2021), temporal knowledge bases for language models (Dhingra et al., 2022), real-time QA benchmarks (Kasai et al., 2024), search-augmented approaches for refreshing model knowledge (Vu et al., 2024), and self-updating present-anchored QA suites (Meem et al., 2024). While these efforts underscore the importance of temporal awareness, they typically treat time sensitivity as a coarse or binary property and do not explicitly characterize how frequently answers change or how complex their temporal dependencies are.

Dataset	Creation	KC	Multi-hop	Recency-Label	Multi-Event	# Ques.
TimeQA (Chen et al., 2021)	Templ.-Wikidata	Wikipedia	✗	✗	✗	20 000
SituatedQA (Zhang and Choi, 2021)	Man.-Filt.	Wikipedia	✗	✗	✗	12 000
TempLama (Dhingra et al., 2022)	Templ./Cloze	Custom-News	✗	✗	✗	50 000
StreamingQA (Liška et al., 2022)	Man.+Gen.	WMT News	✓	✗	✗	410 000
ArchivalQA (Wang et al., 2022)	Gen.	NYT Articles	✗	✗	✗	532 000
ChroniclingAmericaQA (Piryani et al., 2024)	Gen.	Chronicling America Newspapers	✗	✗	✗	485 000
RealTimeQA (Kasai et al., 2024)	News websites	News Articles	✓	✗	✗	~5000
PATQA (Meem et al., 2024)	Templ.-wikidata	Wikipedia	✓	✗	✗	6172
FreshQA (Vu et al., 2024)	Man.	Google Search	✓	✗	✗	600
RecencyQA (*unpublished*)	Man.-Filt.+Gen	Wikipedia/Wikidata	✓	✓	✗	6115
RecencyQA-Multi (*ours*)	Man.-Filt.+Gen	RecencyQA+LLM	✗	✓	✓	1432

Table 1: Overview of question answering datasets. Abbreviations: *Man.*=created manually, *Gen.*=Automatically generated, *Man.-Filt.*=filtered from other datasets, *Man.+Gen.*=created by crowdsourcing and LLM generation *Templ.*=created using templates, *Man.-Filt.+Gen*=filtered from other datasets and LLM generation, *KC*=Knowledge Corpus.

We refer to this requirement as the *recency demand* of a question: the degree to which an answer must be updated over time to remain accurate. Recency demand varies across questions, depending on factors such as the predictability of change and whether an answer depends on a single evolving event or on multiple temporally related events.

A very recent paper, *RecencyQA* (unpublished), introduces a dataset explicitly labeled with recency demands. The authors annotate questions according to how frequently their answers are expected to change and use this dataset to evaluate the ability of various LLMs to predict a question's recency demand. Their results indicate that, while models capture coarse distinctions between static and rapidly changing facts, accurately estimating finer-grained recency requirements remains challenging.

Building on this line of work, our approach extends prior recency-aware QA datasets along several important dimensions. Rather than relying on a fixed set of annotated questions, we introduce a generation-and-labeling pipeline that systematically expands a small seed set into a substantially richer collection of temporal questions. Starting from 75 input questions, the pipeline produces approximately 1500 recency-aware questions by varying both temporal behavior and event structure. In addition to distinguishing stationary from non-stationary temporal dynamics, we explicitly control event dependency—an aspect that has received little explicit attention in prior work—by generating single-event questions as well as multi-event questions involving two or three temporally connected events, with both causal and non-causal relationships. Each generated question is further annotated with a recency label and contextual condition under which the label applies. This structured expansion enables finer-grained analysis of temporal complexity in QA than prior datasets, which typically focus on isolated questions or coarse recency distinctions.

We start explaining the generation pipeline in section 2, followed by creating another pipeline for testing LLMs in section 3. Using this second pipeline we then test various LLMs on the created dataset in section 4. Using the results, we fine-tune a smaller model and compare it to the larger LLMs in section 5. Finally, we conclude with a discussion of findings and future work in section 6.

2 Dataset Creation

Our dataset creation is closely aligned with the *RecencyQA* paper's methodology, but we extend their approach by incorporating multi-event questions and refining the stationarity aspect. As illustrated in Figure 1, the dataset generation pipeline comprises four main stages: question sampling, question generation, recency labeling, and verification conducted by "hand" and an LLM. This pipeline constructs (as described in the

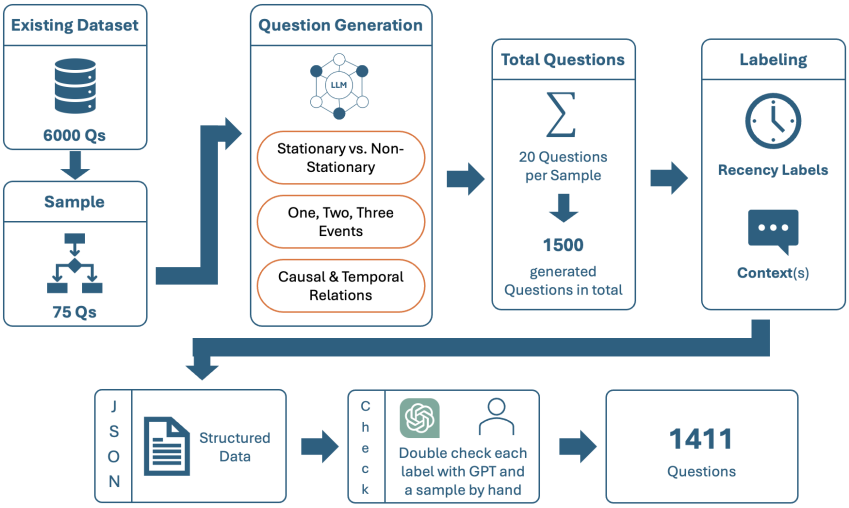

Figure 1: (1) Sampling 75 questions from the RecencyQA dataset. (2) Generating new questions by varying stationarity, number of events, and inter-event relationships. (3) Labeling each question with recency class(es) and corresponding contextual condition(s). (4) Verification of generated questions and labels by "hand" and an LLM. (5) Final dataset with 1411 with improvements as shown in Table 2 and mentioned in Section 2.1.

following sections) a diverse set of recency-aware questions, each annotated with appropriate recency labels and contextual conditions. For this generation and labeling stage, we used the *Llama-3.3-70B-Instruct-Turbo* large language model (Touvron et al., 2023), executed via the Together AI[1] inference platform. The code and dataset are publicly available at GitHub[2] for reproducibility and further research.

2.1 Question Taxonomy

As a core component of the dataset generation pipeline, we organize questions along a small set of structural dimensions, which are stored as explicit metadata for every record.

- **Stationarity**: Whether a questions *recency requirement* is context-invariant. *Stationary* questions keep the same recency label across contexts, whereas *non-stationary* questions change their label when contextual conditions change.

- **Multi-Event**: Questions may depend on a single situation or combine information from multiple distinct or connected events. Multi-event questions increase temporal and structural complexity by requiring the integration of information across several concurrent or related events.

- **Inter-event relationship**: When multiple events are involved, their relationship can be causal—where one event influences another—or purely temporal, where events co-occur in time without an assumed dependency.

- **Recency classes**: Each question is associated with a recency class indicating the expected timescale on which its answer may change (e.g., hours, days, or years). We adopt the recency classes proposed by the *RecencyQA* paper (see Table 2).

- **Contextual conditions**: Recency is not absolute but depends on the assumed state of the world. Contextual conditions describe the situation under which a given recency class applies, allowing the same question (under Non-Stationary) to have different recency labels.

[1] Together AI | The AI Native Cloud
[2] GitHub | RecencyQA_MultiEvent

Recency Class	Expected time until answer change	Example Question
An-Hour	Within an hour	What is the current stock price of Apple?
A-Few-Hours	Within a few hours	What is the current traffic situation on the A9 highway?
A-Day	Within a day	What is todays weather forecast for Munich?
A-Few-Days	Within a few days	What movies are currently trending on Netflix this week?
A-Week	Within a week	What are the top-ranked songs on the Billboard chart this week?
A-Few-Weeks	Within a few weeks	What is the current FIFA world ranking of the German national team?
A-Month	Within a month	What is the current unemployment rate in Italy?
A-Few-Months	Within a few months	What is the current inflation rate in the Eurozone?
A-Year	Within a year	What is the current version of the Java programming language?
A-Few-Years	Within a few years	Who is the president of the United States?
Many-Years	After many years	What is the population of Germany?
Never	Never changes	What is the chemical symbol for gold?

Table 2: The recency classes proposed by the *RecencyQA* paper, along with example questions for each class.

2.2 Question Selection

Having defined the question taxonomy, the next step in the pipeline is to select an initial set of seed questions from which new items are generated. These seeds serve as prompts that anchor topic, style, and temporal structure while still allowing the language model to introduce new events, domains, and cross-event interactions.

In our setting, we randomly select 75 questions from the original *RecencyQA* dataset as seed inputs. The selected questions are evenly distributed across stationarity and recency classes. All questions are provided to the model in a structured `JSON` format (see Appendix A.1).

2.3 Question generation

Given the set of 75 seed questions described in the previous section, the next stage of the pipeline expands each seed into a diverse collection of recency-aware variants that systematically explore the taxonomy introduced in Section 2.1. The goal is not to paraphrase the original questions, but to create new, semantically distinct questions that preserve similar structure while varying event composition and temporal behavior. As mentioned earlier, we utilize the *Llama-3.3-70B-Instruct-Turbo* (Touvron et al., 2023) model for this generation task.

2.3.1 Generation Setup

For each seed question, we invoke a family of prompt templates that condition the model on specific combinations of stationarity and event structure. These templates guide the model to generate questions that are either *stationary* or *non-stationary*, and that depend on either a *single event* or on *multiple* related events. For multi-event questions, we further distinguish between *causal* relationships—where one event influences another—and *temporal-only* relationships, where events merely co-occur in time without direct dependency. To increase task difficulty and better probe temporal reasoning capabilities (see Section 4), we additionally control the structural complexity by generating multi-event questions involving either two or three distinct events, requiring models to integrate information across multiple evolving events.

2.3.2 Prompt Design

Each prompt template is designed to generate two questions per call and explicitly forbids placeholders or paraphrases of the seed, encouraging the generation of genuinely new content rather than surface-level reformulations (Appendix B.1). In practice, however, the language model may very rarely return fewer than two valid questions or fail to produce a usable output. In such cases, only the successfully generated questions are retained.

Table 3 summarizes the ten prompt families used to cover the full space of temporal structures defined by our taxonomy. Each seed question is processed by all ten prompt variants, resulting in up to twenty new questions per seed (two per prompt). Thus our

St.	#Ev.	Rel.	Prompt
S	1	–	B.1.1
S	2	C	B.1.2
S	2	T	B.1.3
S	3	C	B.1.4
S	3	T	B.1.5
NS	1	–	B.1.6
NS	2	C	B.1.7
NS	2	T	B.1.8
NS	3	C	B.1.9
NS	3	T	B.1.10

Table 3: Prompt families used for question generation. **St.** = Stationarity (S = Stationary, NS = Non-stationary), **#Ev.** = number of events, **Rel.** = inter-event relation (C = causal, T = temporal-only).

complete generation process yields up to 1500 new questions, subject to filtering for malformed outputs as described below.

We provide, for each prompt template, representative example questions together with their accompanying contexts and the assigned recency label(s); see Appendix C.1.

2.3.3 Annotation and Metadata

Every generated question is automatically associated with an identifier and annotated with metadata describing its stationarity, event dependency, number of events, and—when applicable—the prompted inter-event relation (causal vs. temporal-only). This process transforms each original question into a family of temporally enriched questions, yielding a dataset that supports fine-grained analysis of how temporal structure, multi-event dependency, and recency demand interact in LLM-based question answering.

2.4 Recency and context labeling

To apply the recency-class dimension introduced in Section 2.1, we annotate each question with one recency class (or two recency classes if non-stationary) that specify on which time scale its correct answer is expected to change. We use the twelve classes proposed by the *RecencyQA* paper (Table 2) and store them as normalized strings (e.g., `An-Hour`, `A-Few-Days`).

Stationary questions receive a single label, while non-stationary questions receive two labels to capture alternative temporal regimes under different world states. For each label, we additionally elicit a one-sentence contextual condition (event, phase, or condition) that makes the question relevant under that regime. The labeling prompt prohibits reasoning and explicit timestamps and is reproduced in Appendix B.3 via the stationary template (Appendix B.3.1) and the non-stationary template (Appendix B.3.2). Labels and conditions are merged into a structured list so downstream models can directly pair each recency class with its motivating scenario. We retain only entries whose labels parse as `JSON` and whose number of conditions matches the expected label count.

The output is stored in a structured `JSON` format (see Appendix A.2) that includes the questions, recency labels, contextual conditions, and the metadata described (Section 2.3.3). In Appendix C.2, we provide a concrete example of a question with its associated labels and contexts in `JSON` format.

2.5 Verification and quality control

To reduce noise from imperfect generations and labels, we apply a two-stage verification procedure combining an LLM-based consistency check with targeted manual review.

First, we used *GPT-5.1 Codex Max* via GitHub Copilot Premium to screen all records. To fit the model's context window, we split the dataset into seven parts and asked the model to validate (i) whether the recency label(s) match the question under the provided contextual condition(s) and (ii) whether the question is sensible and grammatically well-formed, see Appendix B.2 for the exact prompt. The model outputs a list of question IDs flagged as either "correct" or "incorrect".

All items flagged as "incorrect" were then double-checked by hand and removed. Finally, we randomly sampled 75 remaining

Metric	Value
Total questions (all splits)	1411
Unique seed questions	1373
Total Recency Classes	12
Stationary questions	725
Non-stationary questions	686
Single-event questions	286
Two-event questions	555
Three-event questions	570
Multi-event (causal)	565
Multi-event (temporal-only)	560
Avg. question length (tokens)	22.2
Avg. context 1 length (tokens)	11.2
Avg. context 2 length (tokens)	12.6
Total recency labels	2097
Most frequent labels (top-3)	`A-Year` (373), `A-Day` (352), `A-Few-Days` (320)

Table 4: Dataset overview statistics.

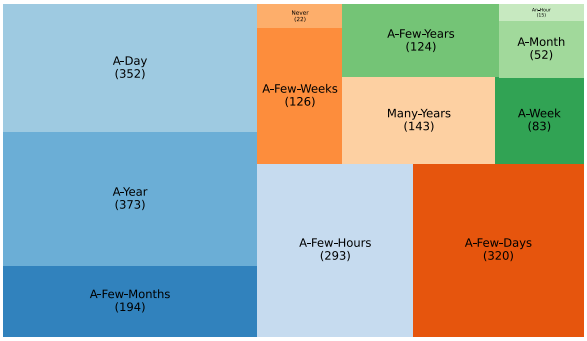

Figure 2: Distribution of recency labels in the final dataset.

questions and audited them manually; only rare edits were necessary (e.g., adjusting a label or removing an ambiguous question).

2.6 Dataset Statistics

We conclude this section by summarizing the resulting dataset after generation, labeling, and verification. Table 4 provides an overview of the key corpus characteristics that we use throughout our experiments and analysis. Figure 2 visualizes the distribution of recency labels across the entire dataset.

3 Testing Pipeline

In this section we describe our testing pipeline for evaluating the performance of language models on our dataset, or on custom datasets. Thus, our pipeline is designed to be usable for any dataset that follows this structure and any LLM that can be accessed via *Together AI*[3]. The pipeline consumes the unmodified `JSON` produced by the generation process (see Appendix A.2).

After loading the dataset, we iterate over every question and unpack its list of temporal contexts and gold labels as provided by the schema in Appendix A.2. Each entry becomes an independent evaluation instance consisting of a question, one context sentence, the corresponding recency label, and metadata describing stationarity, event dependency, and the number of events. This flattening step allows us to handle questions with one or multiple applicable contexts uniformly, while retaining the ability to later aggregate results by stationary vs. non-stationary behavior or by the structural class of the question.

For every instance we assemble the testing prompt shown in Appendix B.4 and submit it to the selected model via Together AI. The prompt enforces single-label answers from the same discrete label set that was used during generation, ensuring direct comparability between model predictions and the human-authored ground truth. The temperature is fixed at 0.0 to minimize randomness and creativity in the outputs.

The pipeline writes one `JSONL` file per model containing all predictions, including the original question, the used context, the gold label, and the predicted label. In addition, it produces a compact summary `JSON` that reports accuracy, a tolerant accuracy (+/- one label), the number of evaluated instances, and the count of invalid responses for each model. These metrics are computed both globally and for slices such as stationary vs. non-stationary or single- vs. multi-event questions, enabling quick inspection of where a model struggles.

[3] One could also use other platforms or options than Together AI, but this would need a small refactoring of how to access and execute those models

4 Results

Building on the question taxonomy from Section 2.1 and the metadata annotations discussed in Section 2.3.3, we evaluate *moonshotai/Kimi-K2-Instruct-0905* (1 Trillion parameters) (Moonshot AI, 2025), *Qwen/Qwen2.5-72B-Instruct-Turbo* (72 Billion parameters) (Team, 2024; Yang et al., 2024) and *deepseek-ai/DeepSeek-V3* (671 Billion parameters) (DeepSeek-AI et al., 2025) with the testing pipeline from Section 3.

4.1 Overall Performance

Qwen2.5-72B attains the strongest accuracy (45.6%) and tolerant F1 accuracy (77.4%). DeepSeek-V3 trails by roughly five absolute points, while the Kimi-K2 instruction model stays below 35% accuracy despite a comparable tolerant score. Invalid generations remain negligible, confirming that the constrained testing prompt in Appendix B.4 keeps responses well-formed.

4.2 Impact of Stationarity

Stationarity is the clearest driver of variance. Each model loses between 13 and 26 percentage points when moving from stationary to non-stationary questions, mirroring the temporal volatility highlighted during annotation (Section 2.3.3). Qwen drops from 62.8% to 36.7% accuracy, indicating that even large instruction models struggle when the required recency hinges on punctual events rather than cyclical updates. Kimi suffers the steepest relative decline (-13.8 points) because its stationary accuracy is already modest, whereas DeepSeek maintains low-30s performance on non-stationary prompts despite exceeding 52% on the stable slice. These gaps confirm that the dataset captures the adaptive reasoning behaviour targeted in Section 2.1.

4.3 Mutli-Event Question Performance

In the final four columns of Table 5, we break down model performance on multi-event questions by number of events and generation type: causal (C) vs. temporal-only (T).

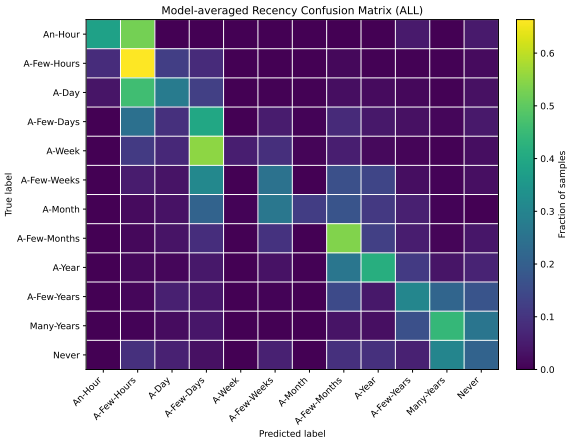

Figure 3: Model-averaged confusion matrix over all 12 recency labels, showing recall per label.

For all three models we don't observe significant performance differences based on the number of events involved in the question. The tolerant accuracy has a slightly more downward trend as the number of events increases (from 80.3% to 66.2% (two events) and 69.3% (three events) (aggregated)). This does indicate that multi-event questions are more difficult than single-event questions, but the number of events itself does not seem to have a strong influence.

When comparing causal vs. temporal-only multi-event questions, we observe that all models perform better on causal questions. Qwen achieves 47.8% accuracy on causal questions compared to 43.5% on temporal-only questions. Kimi and DeepSeek show similar trends with 34.9% vs. 29.8% and 43.9% vs. 35.6%, respectively. This suggests that models may find it easier to reason about cause-and-effect relationships when determining recency, as opposed to purely temporal relationships. Though the differences are not very large, they are consistent across all models.

4.4 Label-wise Behaviour and Error Direction

Per-label statistics expose systematic blind spots. All three systems excel at the shortest horizons (e.g., Qwen reaches recall 55% on `A-Few-Hours` and Kimi exceeds 73%), yet accuracy collapses for intermedi-

Model	Overall		St.		Non-St.		# Events			Multi-event	
	Acc	Tol.	Acc	Tol.	Acc	Tol.	1 Acc	2 Acc	3 Acc	C Acc	T Acc
Kimi	33.0	62.2	42.1	66.3	28.3	60.0	35.0	31.1	33.6	34.9	29.8
Qwen	45.6	77.4	62.8	81.8	36.7	75.1	45.4	44.6	46.6	47.8	43.5
Deepseek	40.8	71.9	52.2	73.2	34.9	71.1	44.9	38.1	41.1	43.9	35.6
Total	39.8	70.5	52.4	73.8	33.3	68.7	41.7	37.9	40.5	42.2	36.3

Table 5: Accuracy (Acc) and tolerant F1 accuracy (Tol., ± 1 label; computed as the tolerant F1 metric). St.: stationary; Non-St.: non-stationary; C: causal; T: temporal-only.

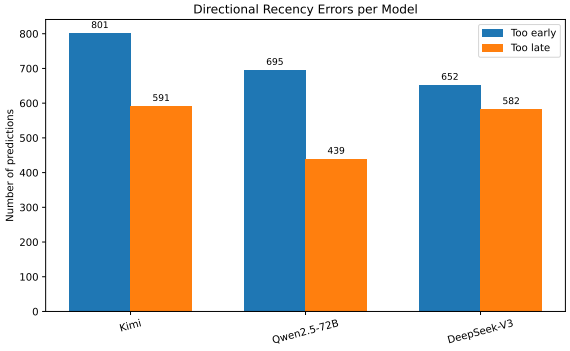

Figure 4: Directional error counts per model, contrasting "too early" vs. "too late" predictions.

ate windows: `A-Week` recall never surpasses 10%, and `A-Month` remains below 20% for Kimi and DeepSeek. Long-horizon labels such as `Many-Years` and `Never` also exhibit low precision (Qwen's best F_1 for `Never` is 6.5%). A condensed, model-averaged confusion heatmap makes these failure modes visible; we therefore refer to Figure 3 for the aggregated confusion matrix over all 12 labels.

Directional errors emphasise another imbalance. Kimi produces 801 "too early" predictions versus 591 "too late", Qwen 695 vs. 439, and DeepSeek 652 vs. 582. The models thus prefer overly fresh information, underestimating how slowly certain questions evolve. This bias matters for downstream systems that rely on recency signals to schedule refreshes: adopting a "check too often" policy could waste computation, whereas missing late updates risks outdated answers. To visualize this effect we refer to Figure 4, a bar chart contrasting the error directions per model.

Finally, tolerant accuracy being between 20 and 30 points higher than accuracy across all models shows that most mistakes deviate by only one label. While encouraging, this plateau also signals that the discrete label borders defined in Section 2.1 remain hard to recover without explicit temporal reasoning, motivating future work on richer reasoning prompts or retrieval-augmented pipelines.

5 LLM Fine-Tuning

In this section we fintune a model and compare it with the other models discussed in Section 4. Therefore we use *Qwen2.5-14B-Instruct-recency* (Yang et al., 2024; Team, 2024) and finetune it via Together AI. We choose this model, because we want to improve the best of the already evaluated models (i.e. Qwen2.5-72B), but we need a smaller model version to be able to finetune it within our compute budget.

5.1 Dataset Splitting

Before finetuning, we split the dataset into a train, test and evaluation part with a $70/15/15$ ratio, using a python script[4]. To prevent data leakage, we split at the *question level*, ensuring that all contextual variants of a question remain within the same partition.

We apply a stratified splitting strategy to preserve the original distribution across the key dataset axes:

(i) event dependency,
(ii) number of events,
(iii) stationarity, and
(iv) generation type.

Stratification labels are constructed as a composite of these attributes and used with

[4] GitHub RecencyQA_MultiEvent

a two-stage *StratifiedShuffleSplit* procedure. Afterwards we flatten the dataset, to make it compatible with the required format of Together AI. We exclusively use the development split for validation during fine-tuning, while the test split keeps in its original hierarchical structure for the evaluation against the other models later on.

5.2 Training

Fine-tuning is performed on the *Qwen2.5-14B-Instruct* base model using parameter-efficient *Low-Rank Adaptation* (LoRA) (Hu et al., 2021) via Together AI. *Supervised fine-tuning* (SFT) with chat-style prompts is applied, framing the task as a single-label classification problem over 12 discrete temporal recency classes.

Each training instance begins with a system instruction that defines the model as an expert in temporal reasoning, followed by a user message containing the question, its context, and the list of admissible labels. The assistant output is restricted to the gold label only, thereby enforcing a strict classification setup.

LoRA adapters are applied to all linear layers with rank $r = 8$ and scaling factor $\alpha = 16$, enabling lightweight adaptation while preserving the base model weights. Training runs for two epochs with a batch size of 8 and a learning rate of 2×10^{-5}. A cosine learning rate scheduler with a warmup ratio of 0.05 is used, and gradient norms are clipped at 1.0. No weight decay is applied.

To ensure consistent training and evaluation conditions, the model is trained deterministically with temperature set to zero. Validation is performed on the held-out development split after each epoch, and the best-performing checkpoint is retained for final evaluation.

5.3 Fine-Tuning Results

We compare the fine-tuned *Qwen2.5-14B-Instruct-recency* (14 billion parameters)(Yang et al., 2024; Team, 2024) model against the models presented in Section4, using the same evaluation pipeline and recency labels. To ensure a fair and direct comparison, all models are re-evaluated on the reduced dataset rather than the full dataset used in the original experiments. Table 6 reports exact accuracy and tolerant F_1 (± 1 label). Despite the large parameter gap (14B vs. 72B–1000B), the fine-tuned model reaches 40.9% accuracy and 69.6% tolerant F_1, essentially matching DeepSeek-V3 (41.4%/69.7%) and outperforming Kimi-K2 (35.5%/64.9%). Qwen2.5-72B remains strongest at 46.5% accuracy and 76.1% tolerant F_1.

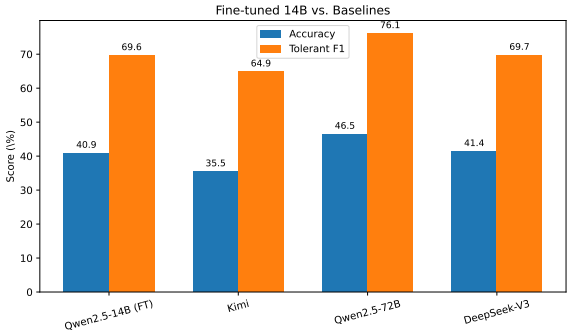

Figure 5: Exact accuracy and tolerant F_1 for the fine-tuned model versus the three baselines.

Stationarity remains the main driver of variance. The fine-tuned model drops from 55.1% to 33.5% accuracy (-21.6 points) between stationary and non-stationary questions. Qwen shows the steepest drop (63.9% to 37.4%), while DeepSeek and Kimi drop by 14.5 and 16.5 points, respectively. For event dependency, the fine-tuned model is nearly invariant in exact accuracy (41.5% single-event vs. 40.7% multi-event), but tolerant F_1 declines from 78.5% to 67.3%, indicating more near-miss errors in multi-event questions.

The fine-tuned model shows the same "too-early" bias as the baselines (128 "too early" vs. 57 "too late").

5.3.1 Comparison between Finetuned and other models

The fine-tuned *Qwen2.5-14B-Instruct-recency* closes most of the gap to much larger models on the reduced test set despite using only 14B parameters (5×

Model	Overall		St.		Non-St.		Single-event		Multi-event	
	Acc	Tol.	Acc	Tol.	Acc	Tol.	Acc	Tol.	Acc	Tol.
Qwen2.5-14B (FT)	40.9	69.6	55.1	72.9	33.5	68.0	41.5	78.5	40.7	67.3
Kimi	35.5	64.9	46.3	70.4	29.8	62.0	29.7	65.6	36.9	64.7
Qwen2.5-72B	46.5	76.1	63.9	82.4	37.4	72.8	46.2	87.7	46.6	73.1
DeepSeek-V3	41.4	69.7	50.9	71.3	36.4	68.9	38.5	70.8	42.2	69.5

Table 6: Accuracy (Acc) and tolerant F_1 (Tol., ± 1 label). St.: stationary; Non-St.: non-stationary.

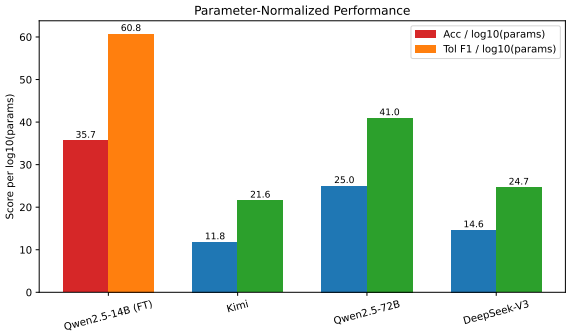

Figure 6: Parameter-normalized performance (score per $\log_{10}$ parameter count), highlighting the fine-tuned models strength per parameter.

smaller than Qwen2.5-72B and orders of magnitude below trillion-parameter systems). In overall accuracy it reaches 40.9%, only 0.5 points below DeepSeek-V3 (41.4%), 5.4 points above Kimi (35.5%), and 5.6 points below Qwen2.5-72B (46.5%). The same pattern holds for tolerant F_1: 69.6% for the fine-tuned model versus 69.7% for DeepSeek, 64.9% for Kimi, and 76.1% for Qwen2.5-72B, indicating that fine-tuning yields performance near the best baselines at a fraction of the size.

By stationarity, the fine-tuned model remains competitive on stationary questions (55.1% accuracy), but degrades on non-stationary questions (33.5%), matching the central difficulty observed across all models. While Qwen2.5-72B still leads in both regimes, fine-tuning narrows the gap to DeepSeek-V3 on non-stationary items (33.5% vs. 36.4%) and stays ahead of Kimi (29.8%). For event dependency, exact accuracy is stable between single- and multi-event settings (41.5% vs. 40.7%), suggesting that fine-tuning improves overall calibration rather than removing the multi-event challenge.

Overall, the fine-tuned model delivers near–state-of-the-art quality at a fraction of the parameter count, making it the most efficient option among the evaluated systems. In particular, it stays competitive with substantially larger models on key metrics, reinforcing that the gains stem from effective adaptation rather than scale alone. This is further emphasized by the parameter-normalized view in Figure 6, where the fine-tuned model achieves the strongest performance per parameter.

6 Summary and Outlook

Having detailed the dataset construction, evaluation, and fine-tuning analysis, we now synthesize the main takeaways.

6.1 Conclusion

We introduced *RecencyQA-Multi*, a systematically generated extension of the original RecencyQA corpus that explicitly varies stationarity, the number of interdependent events, and the relationships between them (Section 2). Starting from only 75 seed questions, our controlled prompting and verification pipeline yielded 1411 questions with fine-grained recency labels and contextual conditions, providing a richer testbed for studying temporal reasoning.

To gauge how well current models exploit this structure, we designed a reusable evaluation pipeline (Section 3) and benchmarked state-of-the-art instruction models (Section 4). Despite tolerant accuracies exceeding 70%, all models misclassify most instances under the strict metric, and accuracy drops sharply for non-stationary and multi-event questions. Qwen2.5-72B performs best overall, yet still struggles to distinguish medium-horizon labels and consistently predicts overly fresh answers, under-

scoring the difficulty of calibrating temporal priors even for large LLMs.

Fine-tuning a smaller model confirms that adaptation can close much of the scale gap: the *Qwen2.5-14B-Instruct-recency* model reaches 40.9% accuracy and 69.6% tolerant F_1 on the reduced test set, matching DeepSeek-V3 and surpassing Kimi-K2 while trailing Qwen2.5-72B. When normalized by parameter count, the fine-tuned model delivers the strongest performance per parameter (Figure 6), underscoring that targeted fine-tuning yields substantial efficiency gains.

6.2 Future Research

Several directions for future work could strengthen both the reliability and expressiveness of recency labels. A natural next step is to construct a dataset in which labels are assigned by humans at scale, ideally with multiple independent annotators per question. This could be facilitated through a lightweight web interface, and the final label could be derived from an aggregation rule such as the median or majority vote to reduce individual bias and noise. This would allow a more accurate evaluation of model performance against human judgment.

In addition, model ensembles offer a complementary avenue: multiple models could be run in parallel and the most consistently predicted label could be used as the final decision. In cases where the ensemble remains inconclusive, additional models (or a second-stage decision process) could be introduced, informed by the earlier model outputs to guide disambiguation. Potentially, this could yield more robust recency estimates by leveraging diverse model perspectives.

Finally, instead of treating recency prediction as a single hard-label task, it may be more appropriate to model ambiguity explicitly by using a distribution over labels as the target. For example, if annotators disagree between "A-Week" and "A-Month", the training signal could reflect this uncertainty via proportional target probabilities. Such soft targets would be less brittle than fixed labels and could better capture the inherently fuzzy boundary between time ranges.

References

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan et al. Language models are few-shot learners, 2020. URL https://arxiv.org/abs/2005.14165.

Wenhu Chen, Xinyi Wang, and William Yang Wang. A dataset for answering time-sensitive questions, 2021. URL https://arxiv.org/abs/2108.06314.

DeepSeek-AI, Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang et al. Deepseek-v3 technical report, 2025. URL https://arxiv.org/abs/2412.19437.

Bhuwan Dhingra, Jeremy R. Cole, Julian Martin Eisenschlos, Daniel Gillick, Jacob Eisenstein et al. Time-aware language models as temporal knowledge bases. *Transactions of the Association for Computational Linguistics*, 10:257–273, 03 2022. ISSN 2307-387X. doi: 10.1162/tacl_a_00459. URL https://doi.org/10.1162/tacl_a_00459.

Kevin Fischer, Darren Fürst, Sebastian Steindl, Jakob Lindner, and Ulrich Schäfer. Question: How do large language models perform on the question answering tasks? answer:, 2024. URL https://arxiv.org/abs/2412.12893.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li et al. Lora: Low-rank adaptation of large language models, 2021. URL https://arxiv.org/abs/2106.09685.

Jungo Kasai, Keisuke Sakaguchi, Yoichi Takahashi, Ronan Le Bras, Akari Asai et al. Realtime qa: What's the answer right now?, 2024. URL https://arxiv.org/abs/2207.13332.

Adam Liška, Tomáš Kočiský, Elena Gribovskaya, Tayfun Terzi, Eren Sezener et al. Streamingqa: A benchmark for adaptation to new knowledge over time in question answering models, 2022. URL https://arxiv.org/abs/2205.11388.

Jannat Meem, Muhammad Rashid, Yue Dong, and Vagelis Hristidis. PAT-questions: A self-updating benchmark for present-anchored temporal question-answering. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar, editors, *Findings of the Association for Computational Linguistics: ACL 2024*, pages 13129–13148, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-acl.777. URL https://aclanthology.org/2024.findings-acl.777/.

Moonshot AI. Kimi-K2-Instruct-0905. https://huggingface.co/moonshotai/Kimi-K2-Instruct-0905, 2025. Accessed: 2026-01-13.

Bhawna Piryani, Jamshid Mozafari, and Adam Jatowt. Chroniclingamericaqa: A large-scale question answering dataset based on historical american newspaper pages. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR 2024, page 2038–2048. ACM, July 2024. doi: 10.1145/3626772.3657891. URL http://dx.doi.org/10.1145/3626772.3657891.

Qwen Team. Qwen2.5: A party of foundation models, September 2024. URL https://qwenlm.github.io/blog/qwen2.5/.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux et al. Llama: Open and efficient foundation language models, 2023. URL https://arxiv.org/abs/2302.13971.

Tu Vu, Mohit Iyyer, Xuezhi Wang, Noah Constant, Jerry Wei et al. FreshLLMs: Refreshing large language models with search engine augmentation. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar, editors, *Findings of the Association for Computational Linguistics: ACL 2024*, pages 13697–13720, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-acl.813. URL https://aclanthology.org/2024.findings-acl.813/.

Jiexin Wang, Adam Jatowt, and Masatoshi Yoshikawa. Archivalqa: A large-scale benchmark dataset for open domain question answering over historical news collections, 2022. URL https://arxiv.org/abs/2109.03438.

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu et al. Qwen2 technical report. *arXiv preprint arXiv:2407.10671*, 2024.

Michael J. Q. Zhang and Eunsol Choi. Situatedqa: Incorporating extra-linguistic contexts into qa, 2021. URL https://arxiv.org/abs/2109.06157.

A JSON-structures

A.1 Input JSON format for the pipeline

```
[
  {
    "q_id": "<string | int>",
    "question": "<string>"
  }
]
```

Figure 7: JSON structure for the RecencyQA_MultiEvent pipeline input.

A.2 Output JSON format from the pipeline

```
[
  {
    "q_id": 3,
    "question": "...",
    "event_dependency": "Single-Event | Multi-Event",
    "num_events": 1 | 2 | 3,
    "generation_type": "causal | temporal_only",   // only for Multi-Event
    "stationary": "YES | NO",
    "labels": [
      {
        "recency_label1": <Recency label>,
        "context1": "short natural-language description of the assumed
           situation"
      },
      {
        "recency_label2": <Recency label>,
        "context2": "short natural-language description of the assumed
           situation"
      }
      // second entry only appears when stationary = NO
    ]
  }
]
```

Figure 8: JSON structure for the RecencyQA_MultiEvent pipeline output.

B Prompt Templates

B.1 Generation prompts

B.1.1 Stationary single-event

```
You generate STATIONARY temporal questions.

Definition (internal):
- A stationary question has a stable temporal update behavior.
- The answer changes over time, but the timespan how often the answer must be updated
  would remain the same regardless of when the question is asked.

Task:
Generate EXACTLY 2 stationary temporal questions focusing on ONE event or process.

Rules:
- Must rely on real-world phenomena that change over time.
- Must be stable, cyclical, predictable, or rhythm-based.
- Must NOT paraphrase the examples.
- Must NOT use placeholders.
- Do NOT mention stationarity in the question.

Examples:
examples

Return JSON:

"questions": ["q1","q2"]
```

B.1.2 Stationary multi-event (causal)

```
You generate STATIONARY multi-event temporal questions.

Task:
Generate EXACTLY 2 stationary temporal questions involving TWO events that are causally
↪  related.

Rules:
- Must involve at least TWO distinct temporal events.
- Events must have a clear cause-effect relationship.
- Temporal behavior must be stable, predictable, cyclical, or regular.
- Must rely on real-world change.
- Do NOT paraphrase examples.
- Do NOT use placeholders.
- Do NOT mention stationarity.

Examples:
examples

Return JSON:

"questions": ["q1","q2"]
```

B.1.3 Stationary multi-event (temporal-only)

```
You generate STATIONARY multi-event temporal questions.

Task:
Generate EXACTLY 2 stationary temporal questions involving TWO events that are ONLY
↪  temporally connected.

Rules:
```

```
- Events must be from clearly different real-world domains.
- Events must NOT influence each other causally.
- Must NOT belong to the same topic, organization, or event series.
- Temporal behavior must be stable, predictable, cyclical, or regular.
- Must rely on real-world change.
- Do NOT paraphrase examples.
- Do NOT use placeholders.
- Do NOT mention stationarity.

Examples:
examples

Return JSON:

"questions": ["q1","q2"]
```

B.1.4 Stationary three-event (causal)

```
You generate STATIONARY multi-event temporal questions.

Task:
Generate EXACTLY 2 stationary temporal questions involving THREE events
that are causally related in a chain or network.

Rules:
- Must involve EXACTLY THREE distinct temporal events.
- Events must have clear cause-effect relationships.
- Temporal behavior must be stable, predictable, cyclical, or regular.
- Must rely on real-world change.
- Do NOT paraphrase examples.
- Do NOT use placeholders.
- Do NOT mention stationarity.

Examples:
examples

Return JSON:

"questions": ["q1", "q2"]
```

B.1.5 Stationary three-event (temporal-only)

```
You generate STATIONARY multi-event temporal questions.

Task:
Generate EXACTLY 2 stationary temporal questions involving THREE events
that are ONLY temporally connected.

Rules:
- Must involve EXACTLY THREE distinct events.
- Events must be from clearly different real-world domains.
- Events must NOT influence each other causally.
- Must NOT belong to the same topic, organization, or event series.
- Temporal behavior must be stable, predictable, cyclical, or regular.
- Must rely on real-world change.
- Do NOT paraphrase examples.
- Do NOT use placeholders.
- Do NOT mention stationarity.

Examples:
examples

```

```
Return JSON:

"questions": ["q1", "q2"]
```

B.1.6 Non-stationary single-event

```
You generate NON-STATIONARY temporal questions.

Definition (internal):
- A non-stationary question has unstable temporal update behavior.
- How frequently the answer must be updated depends strongly on WHEN the question is asked.
- OR the question is only relevant within short, event-dependent windows.

Task:
Generate EXACTLY 2 non-stationary temporal questions focusing on ONE event.

Rules:
- Must rely on quickly evolving or unstable processes.
- Must be meaningful and real-world.
- Do NOT paraphrase examples.
- Do NOT use placeholders.
- Do NOT mention non-stationarity explicitly.

Examples:
examples

Return JSON:

"questions": ["q1","q2"]
```

B.1.7 Non-stationary multi-event (causal)

```
You generate NON-STATIONARY multi-event temporal questions.

Task:
Generate EXACTLY 2 non-stationary temporal questions involving TWO events that are causally related.

Rules:
- At least one event must be unstable, unpredictable, or highly time-sensitive.
- Events must have a clear cause-effect relationship.
- Must rely on real-world temporal change.
- Must involve at least TWO distinct temporal events.
- Do NOT paraphrase examples.
- Do NOT use placeholders.
- Do NOT mention non-stationarity.

Examples:
examples

Return JSON:

"questions": ["q1","q2"]
```

B.1.8 Non-stationary multi-event (temporal-only)

```
You generate NON-STATIONARY multi-event temporal questions.

Task:
```

```
Generate EXACTLY 2 non-stationary temporal questions involving TWO events that are ONLY temporally connected.

Rules:
- Events must be from clearly different real-world domains.
- Events must NOT influence each other causally.
- Must NOT belong to the same topic, organization, or event series.
- At least one event must be unstable, unpredictable, or highly time-sensitive.
- Must rely on real-world temporal change.
- Must involve at least TWO distinct temporal events.
- Do NOT paraphrase examples.
- Do NOT use placeholders.
- Do NOT mention non-stationarity.

Examples:
examples

Return JSON:

"questions": ["q1","q2"]
```

B.1.9 Non-stationary three-event (causal)

```
You generate NON-STATIONARY multi-event temporal questions.

Task:
Generate EXACTLY 2 non-stationary temporal questions involving THREE events
that are causally related.

Rules:
- Must involve EXACTLY THREE distinct temporal events.
- At least one event must be unstable, unpredictable, or highly time-sensitive.
- Events must have clear cause-effect relationships.
- Must rely on real-world temporal change.
- Do NOT paraphrase examples.
- Do NOT use placeholders.
- Do NOT mention non-stationarity.

Examples:
examples

Return JSON:

"questions": ["q1", "q2"]
```

B.1.10 Non-stationary three-event (temporal-only)

```
You generate NON-STATIONARY multi-event temporal questions.

Task:
Generate EXACTLY 2 non-stationary temporal questions involving THREE events
that are ONLY temporally connected.

Rules:
- Must involve EXACTLY THREE distinct temporal events.
- Events must be from clearly different real-world domains.
- Must NOT influence each other causally.
- At least one event must be unstable, unpredictable, or highly time-sensitive.
- Must rely on real-world temporal change.
- Do NOT paraphrase examples.
- Do NOT use placeholders.
- Do NOT mention non-stationarity.
```

```

Examples:
examples

Return JSON:

"questions": ["q1", "q2"]
```

B.2 Verification Prompt

```
Can you please check for every single question in this dataset, if:
- the recency_label1 (and if available recency_label2) is correct for the question and context1 (if available context2)
- and if the questions makes sense in general (gramnatically)
Write the id into correct if correct, otherwise into incorrect.
```

B.3 Labeling prompts

B.3.1 Stationary labeling

```
Analyze this temporal question and produce temporal labels:

"question"

Your tasks (internal reasoning only, output JSON only):

1. Provide ONE recency label:

    "An-Hour": "changes within one hour",
    "A-Few-Hours": "changes within a few hours",
    "A-Day": "changes within one day",
    "A-Few-Days": "changes within a few days",
    "A-Week": "changes within one week",
    "A-Few-Weeks": "changes within a few weeks",
    "A-Month": "changes within one month",
    "A-Few-Months": "changes within a few months",
    "A-Year": "changes within one year",
    "A-Few-Years": "changes within a few years",
    "Many-Years": "changes after 10 or more years",
    "Never": "never changes"


2. Based on the selected label, write a short temporal context (ONE sentence) describing the current event, phase, or condition in which the question is asked. This does not have to be related to the question causally.
   - Must describe an EVENT, PHASE or CONDITION, taking place when the question becomes relevant.
   - No specific years.
   - No meta reasoning.
   - No "current"
   - No "the question is asked"

Return JSON:

"recency_list": ["label1"],
"context_list": ["context1"]
```

B.3.2 Non-stationary labeling

```
Analyze this temporal question and produce temporal labels:

"question"
```

```

Your tasks (internal reasoning only, output JSON only):

1. Provide TWO recency labels (for two different realistic temporal situations), choose from:

    "An-Hour": "changes within one hour",
    "A-Few-Hours": "changes within a few hours",
    "A-Day": "changes within one day",
    "A-Few-Days": "changes within a few days",
    "A-Week": "changes within one week",
    "A-Few-Weeks": "changes within a few weeks",
    "A-Month": "changes within one month",
    "A-Few-Months": "changes within a few months",
    "A-Year": "changes within one year",
    "A-Few-Years": "changes within a few years",
    "Many-Years": "changes after 10 or more years",
    "Never": "never changes"


2. For each selected label provide ONE short temporal context (ONE sentence each) describing the current event, phase, or condition in which the question is asked. This does not have to be related to the question causally.
  - Each context must describe a different EVENT, PHASE or CONDITION, taking place when the question becomes relevant.
   - No specific years.
   - No meta reasoning.
   - No "current"
   - No "the question is asked"

Return JSON:

"recency_list": ["label1", "label2"],
"context_list": ["context1", "context2"]
```

B.4 Testing Prompts

```
You are an expert in temporal reasoning.

Given the following question and its temporal context,
classify the needed recency of the data for the answer.

Question:
question

Context:
context

Choose exactly one label from:
[An-Hour, A-Few-Hours, A-Day, A-Few-Days, A-Week, A-Few-Weeks,
 A-Month, A-Few-Months, A-Year, A-Few-Years, Many-Years, Never]

Answer ONLY with the label.
DO NOT provide any explanations or additional text.
```

C Example Questions

C.1 Specific Questions

Stationary Single Event

Question: What is the current water level in the Amazon River during the wet season?
Context: Heavy rainfall is occurring in the Amazon basin during the wet season.
Recency label: A-Few-Days

Stationary Two Events Causal

Question: Do the daily tidal patterns in the Bay of Fundy trigger the opening and closing of the Annapolis Royal tidal power plant?
Context: Tidal patterns are being closely monitored by researchers at the Bay of Fundy.
Recency label: A-Day

Stationary Two Events Temporal-Only

Question: Are the opening hours of the Louvre Museum in Paris synchronized with the daily tidal patterns in the Bay of Fundy?
Context: Tourist season is in full swing in Europe.
Recency label: Never

Stationary Three Events Causal

Question: Will the rise in global temperatures cause more frequent heatwaves in Australia, which in turn increase the risk of bushfires in the region?
Context: A prolonged period of climate change is being observed globally.
Recency label: Many-Years

Stationary Three Events Temporal-Only

Question: What is the current time in New York when the Tokyo Stock Exchange opens and the first tennis match at Wimbledon begins?
Context: Financial markets and sports events are in full swing during summer mornings.
Recency label: A-Few-Hours

Non-Stationary Single Event

Question: What is the current status of the wildfire in California?
Context 1: Firefighters are working to contain the blaze during intense heat waves.
Context 2: Emergency responders are assessing damage after a night of strong winds.
Recency label 1: A-Few-Hours
Recency label 2: A-Day

Non-Stationary Two Events Causal

Question: How do weather forecasters predict the trajectory of a hurricane after it makes landfall, given the rapid change in atmospheric conditions?
Context 1: Hurricane warnings have been issued for several coastal cities as the storm approaches land.
Context 2: Emergency responders are scrambling to prepare evacuation routes as the hurricane's outer rain bands start to affect the area.
Recency label 1: A-Few-Days
Recency label 2: A-Day

Non-Stationary Two Events Temporal-Only

Question: What was the status of the COVID-19 pandemic in the United States when the Perseverance rover landed on Mars?
Context 1: Scientists are analyzing the latest wave of COVID-19 variants.
Context 2: The world is reflecting on the pandemic's impact during a global health conference.
Recency label 1: A-Few-Months
Recency label 2: A-Year

Non-Stationary Three Events Causal

Question: What impact will the unexpected power outage have on the scheduled software update and the subsequent data backup process, considering the backup window is limited to a narrow time frame?
Context 1: The IT team is rushing to meet a tight project deadline.
Context 2: A severe thunderstorm warning has been issued for the area.
Recency label 1: A-Few-Hours
Recency label 2: A-Day

Non-Stationary Three Events Temporal-Onlys

Question: Does the timing of the annual Monaco Grand Prix overlap with the blooming of cherry blossoms in Japan and the announcement of the Nobel Prize winners?
Context 1: The Monaco Grand Prix is nearing its traditional date in late spring.
Context 2: Tourists are finalizing their travel itineraries for the upcoming cherry blossom festival in Japan.
Recency label 1: A-Year
Recency label 2: A-Few-Days

C.2 Example Question JSON Output


```
{
  "q_id": 12,
  "question": "How many people have been reported injured or missing since the last update on the hurricane landfall in Florida?",
  "event_dependency": "Single-Event",
  "num_events": 1,
  "stationary": "NO",
  "labels": [
    {
      "recency_label1": "A-Few-Hours",
      "context1": "Emergency responders are scrambling to evacuate coastal areas as the hurricane makes landfall."
    },
    {
      "recency_label2": "A-Day",
      "context2": "Relief efforts are underway as communities begin to assess the damage from the storm."
    }
  ]
}
```