

TL;QR: Querying Temporal Discursive Patterns in Textual Corpora

François Hublet¹, Vladimir Krsmanović², Jesús Solano¹,
Ahmed Bouhoula¹, Ana Oliveira da Costa³

¹ETH Zürich, ²CISPA Helmholtz Center for Information Security, ³IST Austria,

Correspondence: francois.hublet@inf.ethz.ch

Abstract

Many questions about large textual corpora concern patterns that unfold over time, such as a political group whose stance on a topic drifts from support to criticism over several years of debates. We present TL;QR (*Too Long; Query with logical temporal Reasoning*), a declarative query language and open-source engine for finding such temporal discursive patterns. In TL;QR, a single query combines natural-language predicates, whose truth is judged by a Large Language Model (LLM), with regular expressions and the operators of a temporal logic. The LLM answers only local questions about individual text passages; a symbolic engine then aggregates these judgments over time and links every match to the recorded passages and predicate decisions behind it. With this division of labor, the temporal and relational reasoning stays exact and auditable even over long time horizons. In addition, when a query includes a regular-expression pre-filter, only a small fraction of the corpus ever reaches the LLM. On a suite of controlled synthetic benchmarks with temporally and semantically complex patterns, TL;QR is substantially more accurate than LLM-only baselines. In a case study on ten years of European Parliament debates (442,593 multilingual speeches), TL;QR identifies candidate shifts in the political groups' stances on the GDPR privacy regulation.

understanding *what* each speech expresses (“this speech criticizes environmental regulation”), tracking *when* speeches occur and in what order (“after months of mostly supportive speeches”), and relating *who* produced them (“speeches by members of the same group”).

Since these questions involve open-ended language understanding, LLMs are a natural starting point as they can readily automate many annotation tasks (Gilardi et al., 2023; Ziemis et al., 2024). The above analysis, however, demands more than just understanding each text in isolation. It also requires reasoning over time and over the relations between texts at different time-points. Annotations must be counted, ordered, and compared across speakers and over months or years. Unfortunately, LLMs are known to be unreliable on such long-horizon tasks (Hsieh et al., 2024; Gao et al., 2024; Chu et al., 2024; Fatemi et al., 2025), and processing entire corpora with them can be costly. In contrast, traditional corpus query tools (Hardie, 2012; Krause and Zeldes, 2016) can transparently and cheaply aggregate annotations but cannot interpret open-ended meaning.

In this paper, we present TL;QR (*Too Long; Query with logical temporal Reasoning*), a neuro-symbolic tool that combines the precision of temporal-logic reasoning with the versatility of natural-language annotations. The user states the pattern of interest as a single declarative query in a formal language, using operators for filtering, classification, counting, comparison, as well as natural language and regular expression patterns. An LLM answers only local questions about individual text fragments, such as “does this passage support environmental regulation?”, while a symbolic engine with a formally defined semantics combines these local answers across time, performing all counting, ordering, and windowing operations. Figure 1 shows this data flow on our running example: a cheap regular-expression filter selects candidate

1 Introduction

Analyzing large textual corpora for complex discursive phenomena, such as stance shifts, recurring rhetorical structures, or argumentative patterns, is central to computational social science, political science, and the digital humanities. Consider a question that a political scientist may ask about a multi-year parliamentary record: *when, if ever, did a political group’s stance on environmental regulation flip from mostly supportive to mostly critical?* Answering it requires three abilities at once:

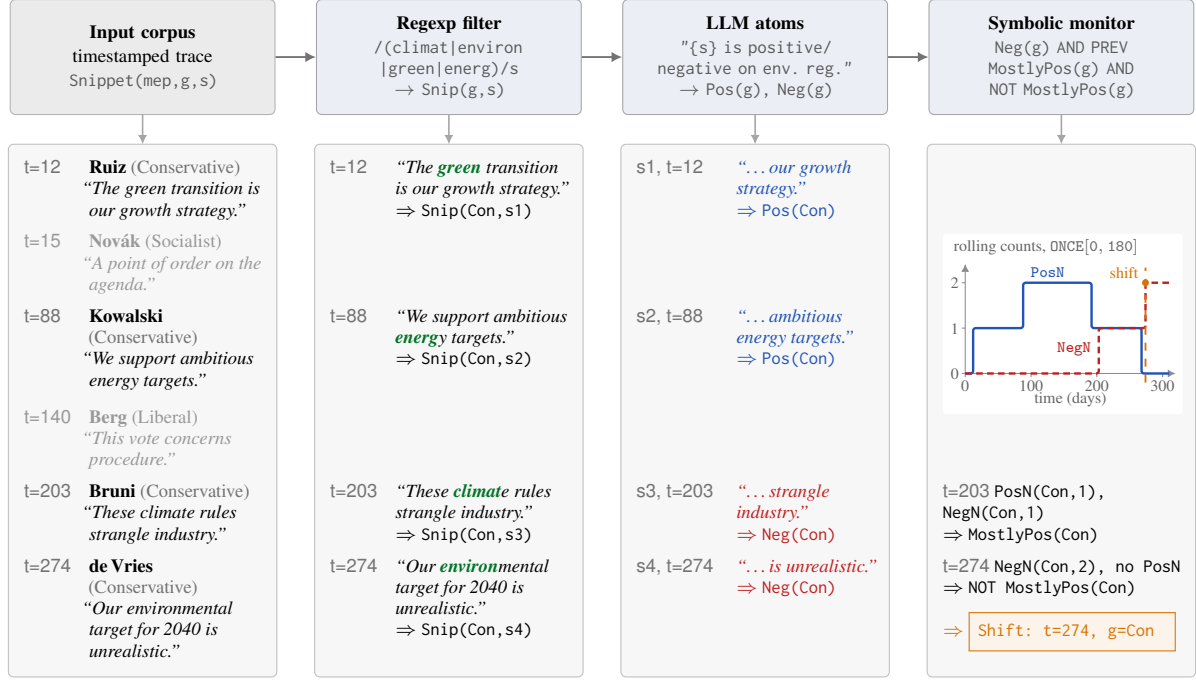


Figure 1: The TL;QR pipeline, illustrated on the query of Figure 2 over a hypothetical corpus with three groups. Each panel shows the intermediate predicate it generates: the regexp pre-filter turns on-topic speeches into Snip events, the LLM atoms label the resulting snippets **positive** or **negative** and emit Pos/Neg, and the symbolic monitor derives the rolling counts PosN/NegN and MostlyPos, and returns the **detected stance shift**. Rows are aligned across the first three panels, so a gap marks a speech the filter dropped. Only the LLM-atoms panel calls a model.

speeches, LLM-backed predicates test each candidate for a supportive and critical stance, and the symbolic engine detects the group whose stance shifts.

The core idea of TL;QR is grounding natural language into atoms in Metric First-Order Temporal Logic (MFOTL) (Chomicki, 1995; Basin et al., 2015b,a). MFOTL acts as the semantic foundation for our specification language, providing operators for quantification, comparisons, and boolean and interval operations. In TL;QR, the atomic predicates of this logic may be corpus events, regular expressions, or natural-language descriptions whose truth is decided by an LLM backend.

To the best of our knowledge, TL;QR is the first query language for text that combines LLM-judged atomic predicates with metric temporal operators, first-order quantification, and aggregations under a formal semantics. The formal semantics gives each query a single unambiguous meaning, avoiding misunderstandings that arise when a pattern is specified in ad hoc code, such as Python, or in imprecise natural language. Furthermore, the use of formally defined semantics and the symbolic reasoning engine provides auditability without requiring additional machinery or external mechanisms.

Every verdict is derived from recorded predicate-level decisions, so each reported match comes with concrete passages and counts, and the user can always trace a verdict back to the annotations used to decide it.

Our contributions are:

1. TL;QR, a specification language that integrates MFOTL with regular-expression and LLM-backed predicates, together with an open-source engine that compiles queries into a hybrid neuro-symbolic monitoring problems and evaluates them incrementally over timestamped corpora (Section 3).
2. A synthetic benchmark suite of temporally and semantically complex discursive phenomena, which provides a controlled testbed for the long-horizon reasoning of LLM pipelines. TL;QR outperforms LLM-only baselines on all benchmarks (Section 4).
3. A case study on ten years of European Parliament debates (442,593 multilingual speeches) demonstrating the scalability of our approach. TL;QR identifies candidate shifts in the political groups' stances on the GDPR for under \$2 of LLM cost; a prompt-only pipeline over

the same corpus would cost orders of magnitude more without providing the same level of transparency and reliability (Section 5).

2 Related Work

LLMs for annotation and aggregation. LLMs are increasingly used as zero- and few-shot annotators for open-ended text analysis, semantic extraction, and annotation in computational social science (Ziems et al., 2024). Their strong performance on these tasks, however, does not carry over to reasoning over long time horizons, e.g., counting and ordering (Gao et al., 2024; Chu et al., 2024). This contrast motivates our design, which uses LLMs only as annotators, while long-horizon reasoning is handled by a symbolic tool.

Corpus query languages. For the analysis of large text corpora, CQPweb (Hardie, 2012) and ANNIS (Krause and Zeldes, 2016) provide scalable and inspectable query systems with user-provided annotations. They support expressive lexical and structural patterns but require manual annotation and cannot process natural-language meanings. Declarative information-extraction and weak-supervision systems such as SystemT (Li et al., 2011), DeepDive (Shin et al., 2015), and Snorkel (Ratner et al., 2017) instead help construct structured annotations or supervision for downstream analysis. More recent semantic data-processing systems incorporate LLMs directly into relational or dataflow pipelines. LOTUS (Patel et al., 2024), Palimpsest (Liu et al., 2025), and DocETL (Shankar et al., 2025) support operations such as semantic filtering, mapping, joining, grouping, and aggregation. As these systems target batch or record-wise processing, they do not support temporal reasoning. TL;QR provides a more expressive specification language, temporal reasoning, and direct support for LLM-judged atoms inside its formulae.

Temporal pattern detection over unstructured streams. Detecting patterns over time and over streams of data is the focus of event processing research. In computer vision, neuro-symbolic video search (Choi et al., 2024; Shah et al., 2026) follows a similar extract-then-reason pattern over frame sequences. Closest to our work is VectorFlow (Chen et al., 2026), which combines LLM-based extraction from document streams with automata for matching. Its pattern operator sup-

ports sequences, conjunction, disjunction, repetition, bounded negation, and temporal windows, and its broader dataflow system provides semantic joins, grouping, and aggregation, similar to TL;QR. Unlike these approaches, TL;QR is grounded in MFOTL, a formal language with a well-defined syntax and semantics that has been used extensively for trace-based monitoring of industrial systems (Basin et al., 2023).

LLMs and symbolic reasoning. Neuro-symbolic research combines neural and symbolic methods to build more robust and powerful systems. Most work in this direction either translates natural language into specifications (Chen et al., 2023; Cosler et al., 2023) or attaches symbolic solvers to improve LLM reasoning (Pan et al., 2023; Olausson et al., 2023). Other approaches, such as Binder (Cheng et al., 2023), Faithful-CoT (Lyu et al., 2023), and PAL (Gao et al., 2023), use LLMs to generate programs and then execute them. We use LLMs differently: in TL;QR they *evaluate predicates* over natural text that the symbolic engine then reasons over, rather than generating programs.

3 TL;QR: Language and Engine

TL;QR consists of two components: a declarative *query language* (Section 3.1) in which the user states the pattern of interest, and a *query engine* (Section 3.2) that evaluates the query efficiently over a large corpus. We introduce both through the running example from the introduction, which we denote by (*): *over a multi-year parliamentary corpus, report each political group whose stance on environmental regulation flips from mostly positive to mostly negative, where “mostly” is measured over a rolling 180-day window.* Note that nothing in our design is specific to parliamentary corpora or stance detection. A query may range over any timestamped collection of texts (social-media streams, forum posts, clinical notes, etc.), and its atoms may encode any annotation task, from topical relevance to rhetorical function. For instance, the benchmarks in Section 4 use the same constructs to detect speakers who evade commitments or coordinate hostile campaigns.

3.1 Query Language

The TL;QR query language extends past-only Metric First-Order Temporal Logic (MFOTL) (Chomicki, 1995) with two new

```

LET Snip(g, s) = EXISTS mep. Snippet(mep, g, s)
AND /(climat|environ|green|energ)/s IN
LET Pos(g) = EXISTS s. Snip(g, s)
AND "{s} is positive on
environmental regulation" IN
LET Neg(g) = EXISTS s. Snip(g, s)
AND "{s} is negative on
environmental regulation" IN
LET PosN(g, x) = x <-
CNT(t; g; ONCE[0,180] (Pos(g) AND tp(t))) IN
LET NegN(g, y) = y <-
CNT(t; g; ONCE[0,180] (Neg(g) AND tp(t))) IN
LET MostlyPos(g) = EXISTS x, y.
PosN(g, x) AND NegN(g, y) AND x >= y IN
Neg(g) AND PREV MostlyPos(g) AND NOT MostlyPos(g)

```

Figure 2: TL;QR query capturing the running example (*), read top-down as a cascade: regexp filter (Snip), LLM stance atoms (Pos/Neg), rolling counts (PosN/NegN), majority comparison (MostlyPos), and the shift condition (last line).

kinds of atomic predicates: natural-language prompts and regular-expression (regexp) predicates. To show how the query language works in practice, a complete query for (*) is shown in Figure 2, and the full syntax in Figure 3. The formal semantics follows Hublet et al. (2025) and is given in Appendix A.

Natural-language predicates. The most distinctive atoms of TL;QR are natural-language predicates: quoted sentences containing placeholders of the form “{x}” that stand for string variables. At evaluation time, an automatic judge (i.e., an LLM backend) decides whether the instantiated sentence is true. In (*), the atom “{s} is positive on environmental regulation” holds for a speech snippet s if and only if the LLM judges this claim to be true of s . Such atoms let queries refer to open-ended semantic phenomena (stance, topical relevance, rhetorical figures, etc.) without any prior annotation of the corpus.

Regex predicates. For surface-level matching that does not require a model call, TL;QR supports regexp predicates of the form $/r/x$, where r is a Python-style regular expression matched against the string variable x . For example, in (*), the atom $/(climat|environ|green|energ)/s$ keeps only snippets that mention the topic at all. In this notation, the trailing s denotes the string variable being matched. The core benefit of regexp predicates is their low computational cost, which makes them a natural complement to the natural-language predicates. This combination enables uses where Regex predicates can be used to filter patterns and pass the content to the LLM only if the con-

tent matches the regex query, avoiding unnecessary token costs.

Data model. Before composing these atoms, we fix how the corpus is represented. A corpus is a *trace*: a sequence of timestamped *databases* $(\tau_1, D_1), \dots, (\tau_n, D_n), \dots$ with $\tau_1 \leq \tau_2 \leq \dots$, where each D_i is a finite set of *events* $e(d_1, \dots, d_k)$ and the index i is called a *time-point*. In our example, every speech snippet contributes an event $\text{Snippet}(mep, g, s)$ recording the speaker, their group g , and the snippet text s . A query ϕ is evaluated at each time-point i and returns all valuations v of its free variables satisfying a sequent $v, i \models \phi$, reading “the variable assignment v satisfies ϕ at time-point i ” (Basin et al., 2015b). In (*), the only free variable is the group g , so each returned valuation names a group whose stance has just shifted.

First-order operators. The Boolean connectives **AND**, **OR**, **NOT** and the existential quantifier **EXISTS** combine atoms, as in first-order logic or in SQL. Built-in arithmetic symbols $(+, *, \dots)$ and user-defined function symbols build terms, and comparison symbols $(=, >=, \dots)$ relate two terms to form atomic formulae. The first definition in Figure 2 uses the connectives and a quantifier: $\text{Snip}(g, s)$ holds when *some* member of group g produced snippet s and s matches the topical regexp.

Temporal operators. Temporal operators relate different time-points using three operators present in TL;QR. **PREV** ϕ holds at i if $i > 1$ and ϕ held at the previous time-point $i - 1$. **ONCE** ϕ holds at i if ϕ held at some time-point $j \leq i$. **SINCE** ϕ holds at i if ϕ held at some $j \leq i$ and ψ held at every time-point after j , up to and including i . An optional interval I bounds how far in the past ϕ may have held: **ONCE** $[0, 180]$ ϕ reads “ ϕ held at some point within the last 180 days”, i.e. $\tau_i - \tau_j \in [0, 180]$. We use this operator to define the rolling window in our running example. Notice that two granularities of time are involved: **PREV** steps back one position in the trace, whereas intervals such as $[0, 180]$ are measured in timestamp units and define intervals. We restrict ourselves to past-only operators because they admit incremental streaming evaluation (Basin et al., 2015b), which our symbolic engine relies on.

Aggregations and LET. Deciding whether a stance is *mostly* positive is a quantitative question, answered by aggregations. An aggregation $y \leftarrow \omega(t; x_1, \dots, x_m; \phi)$ binds y to the aggregate $\omega \in \{\text{CNT}, \text{MIN}, \text{MAX}, \text{AVG}, \text{MED}, \text{SUM}\}$ of the term t

$$\begin{aligned}
t &::= c \mid x \mid f(t, \dots, t) \\
\omega &::= \text{CNT} \mid \text{MIN} \mid \text{MAX} \mid \text{AVG} \mid \text{MED} \mid \text{SUM} \\
I &::= [i, i] \mid [i, *] \\
\phi &::= e(t, \dots, t) \mid \text{NOT } \phi \mid \phi \text{ AND } \psi \mid \phi \text{ OR } \psi \mid \text{EXISTS } x. \phi \\
&\quad \mid \text{PREV } I? \phi \mid \text{ONCE } I? \phi \mid \phi \text{ SINCE } I? \psi \\
&\quad \mid \text{LET } e(x, \dots, x) = \phi \text{ IN } \psi \mid x \leftarrow \omega(t; x, \dots, x; \phi) \\
&\quad \mid t \bowtie t \mid \text{tp}(t) \mid \text{"prompt}(x, \dots, x)\text{"} \mid \text{/regexp}/x
\end{aligned}$$

Figure 3: Syntax of the TL;QR query language. Here e ranges over event and defined predicate names, c over constants, x over variables, f over function symbols, ψ over formulae (like ϕ), i over natural numbers (in timestamp units, e.g. days), and $\bowtie \in \{=, <, <=, >, >=\}$; $I?$ denotes an optional interval, and $\text{tp}(t)$ binds t to the index of the current time-point.

(first position) over all satisfactions of the formula ϕ (third position), grouped by the variables x_i (second position), similar to SQL’s GROUP BY (Basin et al., 2015a). Finally, **LET** introduces named intermediate predicates, so that complex queries can be written modularly and as a combination of simple definitions and queries, enabling easier usage and better overview of the written specification. We can see the usage of these operators in the running example with $\text{PosN}(g, x)$: the body uses the built-in atom $\text{tp}(t)$, which binds t to the index of the current time-point. This means that the body holds for the indices of the time-points within the last 180 days at which group g produced a positive snippet. **CNT** then binds x to the number of such indices.

The complete query. We can look at the example specification shown in Figure 2 and follow how TL;QR assembles these pieces to be able to analyze a large corpus of political speeches. *Snip* is a regexp pre-filter that discards off-topic speeches that are not relevant to our analysis and the next two **LET**s classify the stance of the *already filtered* snippets with LLM atoms, so the number of model calls is bounded by the filtered set rather than by the corpus size. Two **ONCE**[0, 180] aggregations count the recent time-points with positive and with negative snippets per group, and *MostlyPos* compares the counts. Because a grouped aggregation yields no value for an empty group, *MostlyPos*(g) holds only when the window contains at least one snippet of either positive or negative sentiment. The final line of the query fires when a group produces a negative snippet, was mostly positive at the previous time-point, and is no longer mostly positive now. Appendix B lists the full queries used in our experiments.

3.2 Query Engine

A naïve approach to evaluating Natural-language predicates would traverse the formula top-down, calling the LLM every time the logical evaluation reaches a natural-language atom. However, this approach is slow because each model call blocks the evaluation, repeats identical calls, and increases the tool’s runtime cost. To avoid this issue, the TL;QR engine evaluates a query in four stages: (i) compilation of the surface query into a pure-MFOTL formula plus auxiliary Python predicates; (ii) extraction of the non-temporal **LET** prefix of the query; (iii) parallel pre-evaluation of that prefix over all time-points; and (iv) streaming the resulting events into a symbolic monitor that evaluates the temporal part of the query. Figure 1 shows the resulting logical data flow on the query of Figure 2: the panels follow the query’s cascade, with stage (iii) producing the second and third panels and stage (iv) the last one. Our symbolic backend is *EnfGuard* (Hublet et al., 2025), a state-of-the-art monitoring and an enforcement engine for MFOTL that takes an MFOTL formula and a Python file containing function definitions as input.

(i) Compilation. As the symbolic backend of TL;QR does not understand either natural language prompts or regexps, these expressions are compiled into ordinary function symbols. Each distinct natural-language atom “ T ” is assigned a fresh function symbol llm_pred_k and replaced by $\text{llm_pred}_k(x_1, \dots, x_m) = 1$, where the x_i are T ’s placeholders. The compiler binds a Python function to each natural-language atom that calls the configured LLM backend on the instantiated prompt. In our example, the two stance atoms become two such functions. Likewise, regexp atoms compile into re_pred_k , wrapping re.search . Repeated templates are de-duplicated into one function, which enables memoization.

(ii) Non-temporal **LET prefix extraction.** Invoking the LLM from *inside* the monitor would block it on every model call and force serial evaluation. However, the leading **LET**s of a query are typically non-temporal, per-document filters and do not strictly require sequential evaluation. For example, in Figure 2, *Snip*, *Pos*, and *Neg* only look at the current time-point. The engine therefore strips the maximal initial run of **LET**s whose bodies contain no temporal operator, leaving a reduced formula in which they appear as base predicates.

(iii) **Parallel pre-evaluation.** Because the stripped **LETs** are non-temporal, they can be evaluated on each time-point independently, and hence in parallel. The engine analyses them for mutual dependencies (in our example, Pos and Neg depend on Snip) and evaluates them level by level against each local database D_i , dispatching independent **LETs** concurrently. For the same reason, this stage can also run incrementally as new documents arrive. The backend exposes a thread-safe memo cache, so each prompt incurs only a single model call instead of multiple.

(iv) **Streaming symbolic evaluation.** Finally, EnfGuard runs the reduced formula (in our example, the rolling counts, their comparison, and the shift condition) over the trace extended with the pre-computed **LET** events. EnfGuard returns satisfying valuations per time-point. The engine joins them back to the predicate-level decisions to report the evidence behind each match: for $(*)$, the shifting group, the time-point of the shift, and the classified snippets behind the counts. EnfGuard evaluates a well-defined, expressive syntactic fragment of MFOTL and rejects formulae outside it. All queries in this paper lie within this fragment.

4 Evaluation

Our experiments test whether using TL;QR improves accuracy and cost over prompting an LLM directly. Concretely, we ask three questions. **RQ1:** Is TL;QR more accurate than prompt-based baselines on temporally complex queries? **RQ2:** How much of the baseline error comes from temporal reasoning rather than annotation? **RQ3:** How do accuracy and cost depend on the underlying LLM?

4.1 Experimental Setup

Methods. We compare TL;QR to four baselines: KEY-WORDS, ZERO-SHOT, FEW-SHOT and LLM^{LLM}. KEY-WORDS is a purely symbolic baseline that pairs a hand-written lexicon with a hand-written temporal heuristic and makes no LLM calls. ZERO-SHOT and FEW-SHOT LLMs read the raw corpus in chronological batches and output matches directly. LLM^{LLM} is a stronger, diagnostic baseline: a first LLM extracts the same atoms that TL;QR uses, and a second LLM then reasons temporally over the extracted atom stream. Since extraction is shared, any gap between LLM^{LLM} and TL;QR is attributable to the temporal reasoning provided by TL;QR’s symbolic component. All LLM-based

methods run with all three gpt-5.6 variants: luna, terra, and sol.

Metrics. A predicted match is a (group, timestamp) pair. We report F1 over matches that fall within a fixed tolerance of the true timestamp; for STANCE SHIFT, the reference point is the time at which the shift becomes detectable, half a window after it begins. For localisation quality, we report MAE, the Mean Absolute Error (offset in timestamps) between a reported match and the planted event. For cost we count corpus passes weighted by the relative input-token price of the model, in units of one zero-shot pass with the cheapest model (luna); a query with k LLM atoms costs k passes. All numbers are means over three random seeds.

Benchmarks. We construct three synthetic parliamentary benchmarks, each with its own data-generating process, event signature, query, and planted ground truth; the full queries are given in Appendix B. STANCE SHIFT follows groups debating immigration (similar to Figure 2), and a group is reported when it turns from mostly positive to mostly negative, where “mostly” compares rolling counts of positive and negative speeches over a window of 20 time-points. UNRESOLVED HEDGE targets broken commitments: a speaker is reported when they postponed their position on a proposal between 5 and 15 time-points ago and have never committed to that proposal since, so the query combines a metric deadline with the absence of a resolving event. COORDINATED PILE-ON is concerned with concentrated hostility: a target is reported when it accumulates at least 15 hostile claims within a 20-time-point window and exceeds every other target’s count by a margin of 4, with the query comparing aggregates across all actors. Together, the three benchmarks exercise rolling counts, metric deadlines, and cross-group comparison. The generator plants both positive and negative instances, and the corpus text is LLM-written from per-class specifications and cached, so all methods and models see identical corpora. Each benchmark comes in three sizes (1.5k, 3k, and 4.5k events), obtained by scaling the number of actors with fixed time spans and detection windows.

4.2 Results

Accuracy (RQ1). Table 1 shows the full results. TL;QR obtains the best F1 on all three benchmarks, with the largest margins where temporal structure is non-trivial: 62–75% vs. 30–45% on UNRESOLVED

Method	LLM	F1↑ / MAE↓			Cost
		1.5k	3k	4.5k	
STANCE SHIFT					
0-shot	luna	100.0 /0.0	93.3/2.2	60.0/0.0	1×
	sol	100.0 /0.0	93.3/0.0	100.0 /0.0	25×
5-shot	luna	100.0 /0.0	100.0/2.2	93.3/0.0	1×
	sol	100.0 /0.0	100.0 /0.0	100.0 /0.0	25×
LLM ^{LLM}	luna	26.7/0.0	43.3/0.0	86.7/0.0	1×
	terra	16.7/19.0	53.3/1.5	53.3/1.5	10×
	sol	0.0 [†]	66.7/0.0	33.3/0.0	25×
Key-words	—	0.0	0.0	0.0	
TL;QR	luna	100.0 /0.0	100.0 /0.0	100.0 /0.0	2×
	terra	100.0 /0.0	100.0 /0.0	100.0 /0.0	20×
	sol	100.0 /0.0	100.0 /0.0	100.0 /0.0	50×
UNRESOLVED HEDGE					
0-shot	luna	35.6/2.2	29.6/4.0	27.0/3.0	1×
	sol	35.6/0.0	29.6/0.0	45.1/0.0	25×
5-shot	luna	31.7/0.0	14.8/0.0	34.8/0.0	1×
	sol	35.6/0.0	29.6/0.0	45.1/0.0	25×
LLM ^{LLM}	luna	11.1/0.0	12.0/2.2	15.6/2.0	1×
	terra	9.5/9.0	4.2/0.0	14.4/1.1	10×
	sol	0.0	4.2/1.0	5.8/1.0	25×
Key-words	—	22.2/0.0	13.3/0.0	14.3/0.0	
TL;QR	luna	64.4/2.6	75.4 /2.0	73.7 /1.2	2×
	terra	62.6/2.1	67.6/1.6	68.6/0.7	20×
	sol	66.7 /2.1	69.6/0.8	68.6/0.1	50×
COORDINATED PILE-ON					
0-shot	luna	0.0 [†]	0.0	0.0	1×
	sol	16.7/13.0	33.3/10.5	0.0 [†]	25×
5-shot	luna	0.0 [†]	0.0 [†]	0.0 [†]	1×
	sol	33.3/12.0	33.3/11.5	16.7/11.0	25×
LLM ^{LLM}	luna	16.7/10.0	0.0 [†]	0.0	1×
	terra	16.7/10.0	0.0 [†]	0.0 [†]	10×
	sol	33.3/14.3	0.0 [†]	0.0 [†]	25×
Key-words	—	0.0 [†]	0.0	16.7/4.0	
TL;QR	luna	100.0 /0.0	100.0 /0.0	100.0/0.1	1×
	terra	100.0 /0.0	95.2 /0.0	100.0 /0.0	10×
	sol	100.0 /0.0	100.0 /0.0	100.0 /0.0	25×

Table 1: Full results on the three synthetic benchmarks: F1 (%) and MAE (in timestamps), separated by a slash, mean over 3 seeds, for corpus sizes $n \in \{1.5k, 3k, 4.5k\}$. Bold: best F1 per column/benchmark. [†]Recall = 0: the method localised no true instance; precision is undefined, F1 collapses and MAE is omitted.

HEDGE and 95–100% vs. 17–33% on COORDINATED PILE-ON. On STANCE SHIFT, few-shot prompting also reaches 100%, so the benchmark is saturated at this scale. It discriminates between methods only under weaker models (Table 1).

Temporal reasoning versus extraction (RQ2). LLM^{LLM} consumes the same atoms as TL;QR, yet it reaches only 33.3, 3.3, and 11.1 F1 on the three benchmarks. This gap, therefore, isolates temporal reasoning from extraction, showing the

expected LLM weaknesses at counting and ordering over longer time horizons. The results of the symbolic KEY-WORDS baseline are generally poor, with some findings on HEDGE, whose phrasings are lexically distinctive, but none on STANCE, where polarity must be read through negation and attribution that a marker lexicon cannot resolve. MAE further illustrates this: TL;QR localizes stance shifts and pile-ons exactly (0.0) and hedges to within 0.1–2.6 timestamps, whereas every LLM-only baseline that finds a pile-on is off by 10–14 timestamps.

Model sensitivity and cost (RQ3). Across the three backends, TL;QR varies by at most 5 F1 and is identical on two benchmarks, while the baselines swing widely (Table 1). TL;QR’s performance is thus generally robust to the scale of models used for the natural language queries. This robustness to model size further supports the use of cheaper models, further reducing the costs of corpus analysis with TL;QR. The last column of Table 1 prices each configuration in units of a single zero-shot pass over the corpus using the cheapest model. Since TL;QR keeps its accuracy with the cheapest backend, STANCE SHIFT costs 2 units for a perfect score, against 25 units for zero-shot prompting on sol. Our approach provides better or equivalent accuracy at a much lower cost.

5 Case Study: GDPR stances in the European Parliament

To test TL;QR on real-world data, we answer an open question relevant to both political science and privacy studies: *did any parliamentary group’s stance on the General Data Protection Regulation (GDPR) in the European Parliament shift from supportive or neutral to critical over time, and when?*

Setup and query. We use the European Parliament section of ParlLawSpeech (Schwalbach et al., 2025) from 2010 onwards: 442,593 multilingual plenary speeches by twelve political groups, covering most of the GDPR’s parliamentary history from the 2012 proposal through 2016 adoption to post-2018 application. One formula describes the pattern in four stages. Firstly, a regexp atom matches only the speeches that name the regulation in any official language (GDPR, DSGVO, RGD, ...). Afterward, a natural-language atom retains only those genuinely discussing data protection, filtering out those that only use the acronym in passing or in unrelated contexts. The remaining two steps

Predicate	IAA		gpt-5.6-luna vs. human					
	<i>n</i>	Agr.	κ	Acc.	P	R	F1	κ
Relevance	50	100	1.0	.96	1.0	.937	.967	.920
Positive stance	50	80	.598	.75	.720	.958	.822	.600
Negative stance	50	96	.913	.96	.680	1.0	.810	.680

Table 2: Human validation of the LLM predicates. Inter-annotator agreement (IAA) columns report raw agreement and Cohen’s κ between the two primary annotators. Accuracy, precision, recall, F1, and Cohen’s κ are computed with respect to the adjudicated gold labels on the stratified evaluation and reweighted based on class sizes.

classify supportive versus critical stances, and temporal operators detect the first point at which a group’s negative count overtakes its positive count within a rolling 180-day window. gpt-5.6-luna was used as the LLM backend. The query’s structure is thus similar to Figure 2. The full query and prompts are shown in Appendix B.

Pre-filtering. The filtering makes the query affordable: the regexp reduces 442,593 speeches to 343 potential GDPR mentions. Next, the relevance atom identifies 186 speeches that genuinely discuss the regulation; only those reach the stance classifier. An LLM-only pipeline would read all 442,593 speeches, whereas our filtering reduces this by three orders of magnitude and can be expressed as part of the query without the need for an external filtering mechanism.

Cost and runtime. The analysis issues 715 LLM calls (343 relevance judgments plus two stance atoms for each of the 186 survivors) in approximately 15 minutes and at a cost of $\sim$ \\$1. Most computation time is spent on regexp evaluation and model calls at 16-way concurrency.

Findings. Of the 186 relevant speeches, 147 are classified as supportive and 26 as critical. The query reveals five shifts by the center-left Greens/EFA in December 2019, the center-right EPP in April 2016 and October 2020, and the national-conservative ECR in February 2020 and May 2022 (see Figure 5). We can correlate these shifts with real-life events, with the first EPP shift following the Council’s adoption of the text and subsequent shifts occurring during the regulation’s implementation phase. Three groups with a supportive stance, the GUE/NGL, S&D, and Renew groups, and a far-right group with a mixed attitude, the ID group, register no shift. Each verdict produced by TL;QR carries classified speeches as witnesses, so the ex-

planation of how each verdict was made can be inspected. After inspecting the evidence, the potential Greens/EFA shift—a single speaker voicing doubts about implementation effectiveness—does not appear to mark a qualitative departure from the group’s overall supportive stance. In contrast, all other potential shifts feature sustained criticism of the law. One also notes shares of critical statements approaching or touching 50% by the Renew and EPP groups on several occasions, revealing a mixed attitude of center-right members especially in the immediate aftermath of GDPR’s entry into force.

Human Validation. In Table 2, we validate the accuracy of our natural-language atoms against human annotations. For each predicate, 50 speeches were selected by stratified random sampling with balanced numbers of positive and negative predictions. Two authors independently labeled all sampled speeches, with disagreements resolved by a third. The adjudicated labels constitute the gold standard. We find weighted F1 score and recall above 0.8 and 0.9, respectively, for gpt-5.6-luna. Since recall is high, expert review of positive annotations can be used at a relatively low cost to further reduce potential inaccuracies.

6 Conclusion

We have presented TL;QR, a declarative query language and open-source engine for querying temporal discursive patterns in large timestamped corpora. The core novelty of our system is its complementary use of neural and symbolic tools. An LLM is used as a judge that only processes local predicates about individual pieces of text, while a symbolic engine with a formal semantics performs all counting, ordering, and windowing. Our experiments show that TL;QR is more accurate than LLM-only baselines and that its advantage is due to exact temporal reasoning. Additionally, TL;QR provides good accuracy even with an unexpensive LLM backend, whereas the baselines may degrade sharply.

Future work will extend support to future operators and richer atoms, such as retrieval-based or cross-document predicates. Currently, TL;QR treats every LLM judgment as a Boolean. Since the accuracy of each atom can be measured on a labeled sample, a natural extension is to propagate these measurements through the formula, so that every verdict carries a calibrated confidence instead of a binary truth value.

7 Limitations

Atom quality bounds performance. TL;QR’s temporal layer is exact, but its verdicts are only as reliable as the LLM judgments they rest on. We can observe this on our benchmarks, where, when the monitor is given the ground-truth atoms, it recovers the planted pattern perfectly, so the loss that TL;QR does incur in Table 1 comes entirely from atom errors. Moreover, because queries are temporal, a single atom error can affect verdicts at later time-points, for instance, by starting or ending a rolling majority too early.

Query and prompt authoring. Writing a TL;QR query requires familiarity with structured query languages, roughly at the level of a proficient SQL user, together with some comfort with the temporal operators. We provide a reusable four-step recipe, but we have not run a user study, and TL;QR offers no automated translation from a natural-language description into a query. The natural-language atoms themselves must also be phrased with care: the case-study prompts in Appendix B each run to a paragraph.

Synthetic benchmarks. Our corpora are generated from per-class specifications, so their text is less varied than real parliamentary speech and their ground truth is correct by construction rather than by annotation. Absolute scores should therefore be read as relative comparisons between methods under a controlled generator, not as estimates of performance on real-world tasks.

The case study is exploratory. The GDPR analysis reports *candidate* shifts from a single corpus and a single query. The validation in Table 2 shows high recall but only moderate precision for the two stance atoms, so individual classifications require expert review before any conclusion about European politics is drawn; the Greens/EFA verdict, which rests on a single speaker and does not survive inspection, illustrates why. In addition, the regexp pre-filter keeps only speeches that name the regulation explicitly, so speeches that discuss it in other words are missed silently, and the cost advantage over a prompt-only pipeline is an estimate based on token prices.

Engine constraints. The symbolic backend (Hublet et al., 2025) evaluates a syntactic fragment of MFOTL: only past-time temporal operators are supported, and some formulae, such

as negated operands under temporal operators, are approximated rather than decided exactly and must therefore be avoided. Queries outside this fragment have to be rewritten, as the negation-free form of the unresolved-hedge query in Appendix B illustrates.

8 Ethical considerations

Incorrect classification of public speeches can misrepresent the political stance of individuals. Since the findings of our case study are only preliminary, we do not discuss individual speakers’ detected stances and report only group-level outcomes, underlining the need for further qualitative analysis to reach substantive conclusions.

9 Licence for artifacts

EnfGuard is released under the GNU GPLv3 licence. The ParlLawSpeech dataset is released under the CC BY 4.0 licence. We release our own software under GNU GPLv3.

10 Acknowledgments

This work was partially supported by the European Research Council (ERC) Grant HYPER (No. 101055412). Views and opinions expressed are, however, those of the authors only and do not necessarily reflect those of the European Union or the European Research Council Executive Agency. Neither the European Union nor the granting authority can be held responsible for them.

References

- David Basin, Felix Klaedtke, Srdjan Marinovic, and Eugen Zălinescu. 2015a. Monitoring of temporal first-order properties with aggregations. *Formal methods in system design*, 46(3):262–285.
- David A. Basin, Daniel Stefan Dietiker, Srdan Krstic, Yvonne-Anne Pignolet, Martin Raszyk, Joshua Schneider, and Arshavir Ter-Gabrielyan. 2023. [Monitoring the internet computer](#). In *Formal Methods - 25th International Symposium, FM 2023, Lübeck, Germany, March 6-10, 2023, Proceedings*, volume 14000 of *Lecture Notes in Computer Science*, pages 383–402. Springer.
- David A. Basin, Felix Klaedtke, Samuel Müller, and Eugen Zălinescu. 2015b. [Monitoring metric first-order temporal properties](#). *J. ACM*, 62(2):15:1–15:45.
- Shu Chen, Junhan Liu, Deepti Raghavan, and Ugur Cetintemel. 2026. Vectraflow: Long-horizon semantic processing over data and event streams with llms. *arXiv preprint arXiv:2604.03855*.

- Yongchao Chen, Rujul Gandhi, Yang Zhang, and Chuchu Fan. 2023. [NL2TL: transforming natural languages to temporal logics using large language models](#). In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, EMNLP 2023, Singapore, December 6-10, 2023*, pages 15880–15903. Association for Computational Linguistics.
- Zhoujun Cheng, Tianbao Xie, Peng Shi, Chengzu Li, Rahul Nadkarni, Yushi Hu, Caiming Xiong, Dragomir Radev, Mari Ostendorf, Luke Zettlemoyer, Noah A. Smith, and Tao Yu. 2023. [Binding language models in symbolic languages](#).
- Minkyu Choi, Harsh Goel, Mohammad Omama, Yunhao Yang, Sahil Shah, and Sandeep Chinchali. 2024. [Neuro-symbolic video search](#). *CoRR*, abs/2403.11021.
- Jan Chomicki. 1995. [Efficient checking of temporal integrity constraints using bounded history encoding](#). *ACM Trans. Database Syst.*, 20(2):149–186.
- Zheng Chu, Jingchang Chen, Qianglong Chen, Weijiang Yu, Haotian Wang, Ming Liu, and Bing Qin. 2024. [Timebench: A comprehensive evaluation of temporal reasoning abilities in large language models](#). In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2024, Bangkok, Thailand, August 11-16, 2024*, pages 1204–1228. Association for Computational Linguistics.
- Matthias Cosler, Christopher Hahn, Daniel Mendoza, Frederik Schmitt, and Caroline Trippel. 2023. [nl2spec: Interactively translating unstructured natural language to temporal logics with large language models](#). In *Computer Aided Verification - 35th International Conference, CAV 2023, Paris, France, July 17-22, 2023, Proceedings, Part II*, Lecture Notes in Computer Science, pages 383–396. Springer.
- Bahare Fatemi, Mehran Kazemi, Anton Tsitsulin, Karishma Malkan, Jinyeong Yim, John Palowitch, Sungyong Seo, Jonathan Halcrow, and Bryan Perazzi. 2025. [Test of time: A benchmark for evaluating llms on temporal reasoning](#). In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net.
- Luyu Gao, Aman Madaan, Shuyan Zhou, Uri Alon, Pengfei Liu, Yiming Yang, Jamie Callan, and Graham Neubig. 2023. [PAL: Program-aided language models](#). In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 10764–10799. PMLR.
- Muhan Gao, Taiming Lu, Kuai Yu, Adam Byerly, and Daniel Khashabi. 2024. [Insights into LLM long-context failures: When transformers know but don't tell](#). In *Findings of the Association for Computational Linguistics: EMNLP 2024, Miami, Florida, USA, November 12-16, 2024*, Findings of ACL, pages 7611–7625. Association for Computational Linguistics.
- Fabrizio Gilardi, Meysam Alizadeh, and Maël Kubli. 2023. [Chatgpt outperforms crowd-workers for text-annotation tasks](#). *CoRR*, abs/2303.15056.
- Andrew Hardie. 2012. Cqpweb—combining power, flexibility and usability in a corpus analysis tool. *International journal of corpus linguistics*, 17(3):380–409.
- Cheng-Ping Hsieh, Simeng Sun, Samuel Kriman, Shantanu Acharya, Dima Rekesh, Fei Jia, Yang Zhang, and Boris Ginsburg. 2024. [RULER: what's the real context size of your long-context language models?](#) *CoRR*, abs/2404.06654.
- François Hublet, Leonardo Lima, David A. Basin, Srđan Krstić, and Dmitriy Traytel. 2025. [Scaling up proactive enforcement](#). In *Computer Aided Verification - 37th International Conference, CAV 2025, Zagreb, Croatia, July 23-25, 2025, Proceedings, Part III*, volume 15933 of *Lecture Notes in Computer Science*, pages 370–392. Springer.
- Thomas Krause and Amir Zeldes. 2016. [ANNIS3: A new architecture for generic corpus query and visualization](#). *Digit. Scholarsh. Humanit.*, 31(1):118–139.
- Yunhao Li, Frederick Reiss, and Laura Chiticariu. 2011. Systemt: A declarative information extraction system. In *Proceedings of the ACL-HLT 2011 System Demonstrations*, pages 109–114.
- Chunwei Liu, Matthew Russo, Michael Cafarella, Lei Cao, Peter Baile Chen, Zui Chen, Michael Franklin, Tim Kraska, Samuel Madden, Rana Shahout, and 1 others. 2025. Palimpsest: Optimizing ai-powered analytics with declarative query processing. In *Proceedings of the Conference on Innovative Database Research (CIDR)*, page 2.
- Qing Lyu, Shreya Havaldar, Adam Stein, Li Zhang, Delip Rao, Eric Wong, Marianna Apidianaki, and Chris Callison-Burch. 2023. [Faithful chain-of-thought reasoning](#). In *Proceedings of the 13th international joint conference on natural language processing and the 3rd conference of the Asia-Pacific Chapter of the Association for Computational Linguistics (Volume 1: Long papers)*, pages 305–329.
- Theo Olausson, Alex Gu, Benjamin Lipkin, Cedegao E. Zhang, Armando Solar-Lezama, Joshua B. Tenenbaum, and Roger Levy. 2023. [LINC: A neurosymbolic approach for logical reasoning by combining language models with first-order logic provers](#). In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, EMNLP 2023, Singapore, December 6-10, 2023*, pages 5153–5176. Association for Computational Linguistics.
- Liangming Pan, Alon Albalak, Xinyi Wang, and William Yang Wang. 2023. [Logic-lm: Empowering large language models with symbolic solvers for](#)

faithful logical reasoning. In *Findings of the Association for Computational Linguistics: EMNLP 2023, Singapore, December 6-10, 2023*, Findings of ACL, pages 3806–3824. Association for Computational Linguistics.

Liana Patel, Siddharth Jha, Melissa Pan, Harshit Gupta, Parth Asawa, Carlos Guestrin, and Matei Zaharia. 2024. Semantic operators: a declarative model for rich, ai-based data processing. *arXiv preprint arXiv:2407.11418*.

Alexander Ratner, Stephen H Bach, Henry Ehrenberg, Jason Fries, Sen Wu, and Christopher Ré. 2017. Snorkel: Rapid training data creation with weak supervision. In *Proceedings of the VLDB endowment. International conference on very large data bases*, volume 11, page 269.

Jan Schwalbach, Lukas Hetzer, Sven-Oliver Proksch, Christian Rauh, and Miklós Sebők. 2025. Parllawspeech. (Version 1.0.0) [Data set]. GESIS, Köln. <https://doi.org/10.7802/2824>.

Sahil Shah, S. P. Sharan, Harsh Goel, Minkyu Choi, Mustafa Munir, Manvik Pasula, Radu Marculescu, and Sandeep Chinchali. 2026. **Neus-qa: Grounding long-form video understanding in temporal logic and neuro-symbolic reasoning.** In *Fortieth AAAI Conference on Artificial Intelligence, Thirty-Eighth Conference on Innovative Applications of Artificial Intelligence, Sixteenth Symposium on Educational Advances in Artificial Intelligence, AAAI 2026, Singapore, January 20-27, 2026*, pages 8805–8813. AAAI Press.

Shreya Shankar, Tristan Chambers, Tarak Shah, Aditya G. Parameswaran, and Eugene Wu. 2025. **Docetl: Agentic query rewriting and evaluation for complex document processing.** *Proc. VLDB Endow.*, 18(9):3035–3048.

Jaeho Shin, Sen Wu, Feiran Wang, Christopher De Sa, Ce Zhang, and Christopher Ré. 2015. Incremental knowledge base construction using deepdive. In *Proceedings of the VLDB Endowment International Conference on Very Large Data Bases*, volume 8, page 1310.

Caleb Ziems, William Barr Held, Omar Shaikh, Jiaao Chen, Zhehao Zhang, and Diyi Yang. 2024. **Can large language models transform computational social science?** *Comput. Linguistics*, 50(1):237–291.

A MFOTL semantics

Let us fix a trace $\sigma = (\tau_1, D_1), \dots, (\tau_n, D_n)$ and interpretations $c^{\mathcal{I}}$ and $f^{\mathcal{I}}$ of all constant and function symbols, as well as a Boolean prompt evaluator LLM. Temporal operators without an interval attached are treated as carrying the interval $[0, \infty)$. The semantics of TL;QR are given in Figure 4.

$$\begin{aligned}
v, i \models e(t_1, \dots, t_k) &\iff e(\llbracket t_1 \rrbracket_v, \dots, \llbracket t_k \rrbracket_v) \in D_i \\
v, i \models t_1 \bowtie t_2 &\iff \llbracket t_1 \rrbracket_v \bowtie \llbracket t_2 \rrbracket_v \\
v, i \models \text{tp}(t) &\iff \llbracket t \rrbracket_v = i \\
v, i \models \text{NOT } \phi &\iff v, i \not\models \phi \\
v, i \models \phi \text{ AND } \psi &\iff v, i \models \phi \wedge v, i \models \psi \\
v, i \models \phi \text{ OR } \psi &\iff v, i \models \phi \vee v, i \models \psi \\
v, i \models \text{EXISTS } x. \phi &\iff \exists d. v[x \mapsto d], i \models \phi \\
v, i \models \text{PREV } I \phi &\iff i > 1 \wedge \tau_i - \tau_{i-1} \in I \wedge v, i-1 \models \phi \\
v, i \models \text{ONCE } I \phi &\iff \exists j \leq i. \tau_i - \tau_j \in I \wedge v, j \models \phi \\
v, i \models \phi \text{ SINCE } I \psi &\iff \exists j \leq i. \tau_i - \tau_j \in I \wedge v, j \models \psi \\
&\quad \wedge \forall k (j < k \leq i \Rightarrow v, k \models \phi) \\
v, i \models \text{LET } e(\bar{x}) = \phi \text{ IN } \psi &\iff v, i \models_{\sigma'} \psi \text{ where} \\
&\quad \sigma' = \sigma[e(\bar{d}) \in D_j \iff v[\bar{x} \mapsto \bar{d}], j \models \phi] \\
v, i \models x \leftarrow \omega(t; \bar{y}; \phi) &\iff v(x) \in \omega(M) \text{ where } \bar{z} = \text{fv}(\phi) \setminus \bar{y} \\
&\quad \text{and } M = \{ \llbracket t \rrbracket_{v[\bar{z} \mapsto \bar{d}]} \mid \\
&\quad v[\bar{z} \mapsto \bar{d}], i \models \phi, \bar{d} \in \mathcal{D}^{|\bar{z}|} \} \\
&\quad \text{and } |\bar{y}| > 0 \Rightarrow M \neq \emptyset \\
v, i \models \text{"prompt"}(\bar{x}) &\iff \text{LLM}(\text{"prompt"}(\bar{x})[\bar{x} \mapsto v(\bar{x})]) \\
v, i \models /x/x &\iff v(x) \text{ matches the regexp } x \\
\llbracket c \rrbracket_v &= c^{\mathcal{I}}, \quad \llbracket x \rrbracket_v = v(x), \\
\llbracket f(t_1, \dots, t_k) \rrbracket_v &= f^{\mathcal{I}}(\llbracket t_1 \rrbracket_v, \dots, \llbracket t_k \rrbracket_v)
\end{aligned}$$

Figure 4: Semantics of TL;QR. σ is the signature interpretation, $\mathcal{D}$ the domain, and $\llbracket t \rrbracket_v$ the value of term t under valuation v .

B Queries Used in the Experiments

We give the four queries evaluated in Sections 4 and 5. Natural-language atoms are shown in full for the synthetic benchmarks; the case-study prompts are abbreviated, as each runs to a paragraph.

B.1 Synthetic Benchmarks

Stance shift. A group is reported when it speaks negatively, was mostly positive one time-point earlier, and is mostly negative now, where “mostly” compares rolling counts over a window of 20 time-points.

```

LET Pos(g) = EXISTS mep, text. Speech(mep,g,text)
AND "{text} expresses a positive, welcoming, or
humanitarian stance towards immigration" IN
LET Neg(g) = EXISTS mep, text. Speech(mep,g,text)
AND "{text} expresses a negative, hostile, or
restrictive stance towards immigration" IN
LET MostlyPos(g) = EXISTS x, y.
x <- CNT(t; g; ONCE[0,20] (Pos(g) AND tp(t))) AND
y <- CNT(t; g; ONCE[0,20] (Neg(g) AND tp(t))) AND
x > y IN
LET MostlyNeg(g) = EXISTS x, y.
x <- CNT(t; g; ONCE[0,20] (Pos(g) AND tp(t))) AND
y <- CNT(t; g; ONCE[0,20] (Neg(g) AND tp(t))) AND
x < y IN
Neg(g) AND (ONCE[1,1] MostlyPos(g))
AND MostlyNeg(g)

```

Unresolved hedge. A speaker is reported when they hedged on some proposal between $d=5$ and

$d+k=15$ time-points ago and have not committed to that same proposal anywhere in the lookback.

```
LET Hedge(sp, prop) = EXISTS text.
  Utterance(sp, prop, text) AND
  "{text} contains an explicit statement postponing
  or withholding a position on the proposal" IN
LET Commit(sp, prop) = EXISTS text.
  Utterance(sp, prop, text) AND
  "{text} contains an explicit pledge to support or
  vote for the proposal, in the speaker's own
  voice" IN
EXISTS prop. (ONCE[5,15] Hedge(sp, prop))
AND (NOT (ONCE[0,15] Commit(sp, prop)))
```

This is a negation-free rendering of the pattern, which is naturally stated with **SINCE**. The two coincide on this generator, because a resolving commitment is always planted within d of its hedge and therefore falls inside the $[0, d+k]$ lookback.

Coordinated pile-on. A target is reported when its count of hostile claims over a 20-time-point window reaches $\theta=15$ and exceeds every other target's count by a margin of 4.

```
LET Att(tg) = EXISTS sp, text. Claim(sp, tg, text)
AND "{text} makes a hostile or accusatory claim
against the person it refers to" IN
LET Cnt(tg, n) =
  n <- CNT(t; tg;
  ONCE[0,20] (Att(tg) AND tp(t))) IN
LET Cnt2(tg2, m) =
  m <- CNT(t; tg2;
  ONCE[0,20] (Att(tg2) AND tp(t))) IN
EXISTS n. Cnt(tg, n) AND n >= 15 AND
  (NOT (EXISTS tg2, m. Cnt2(tg2, m) AND
  (NOT (tg2 = tg) AND m + 4 > n)))
```

The margin makes the comparison robust to atom noise: a bare “no other target reaches this count” is decided by a single count, so a few false-positive atoms on one bystander flip the verdict.

B.2 Case Study

The GDPR query stages a regexp filter, a relevance atom, two stance atoms and a rolling-window comparison. For readability, the four atom bodies are written here as the placeholders G_{dprRe} , P_{rel} , P_{pos} and P_{neg} ; their full texts follow the query.

```
LET Mention(mep, g, text) = Speech(mep, g, text)
AND /GdprRe/text IN
LET Rel(mep, g, text) = Mention(mep, g, text)
AND "P_rel" IN
LET Pos(g) = EXISTS mep, text. Rel(mep, g, text)
AND "P_pos" IN
LET Neg(g) = EXISTS mep, text. Rel(mep, g, text)
AND "P_neg" IN
LET CntPos(g, x) =
  (x <- CNT(t; g; ONCE[0,180] (Pos(g) AND tp(t))))
OR (x = 0 AND NOT (EXISTS t. ONCE[0,180] Pos(g))) IN
LET CntNeg(g, x) =
  (x <- CNT(t; g; ONCE[0,180] (Neg(g) AND tp(t))))
OR (x = 0 AND NOT (EXISTS t. ONCE[0,180] Neg(g))) IN
LET NegFrac(g, r) = EXISTS p, n.
  (ONCE[0,180] (Pos(g) OR Neg(g)) AND CntPos(g, p)
  AND CntNeg(g, n) AND p + n > 0;
  r <- (1000 * n) / (p + n)) IN
LET MostlyNeg(g) =
  EXISTS r. NegFrac(g, r) AND r > 500 IN
ONCE[0,180] (Pos(g) OR Neg(g)) AND MostlyNeg(g)
AND PREV (NOT MostlyNeg(g))
```

Note the syntax $\phi; x \leftarrow t$, which is syntactic sugar for $x \leftarrow \text{AVG}(t; y_1, \dots, y_n; \phi)$, where $y_1, \dots, y_n$ are the free variables of ϕ . This behaves exactly like a traditional assignment.

Timestamps are days, so the **ONCE**[0, 180] windows are six calendar months. $CntPos$ and $CntNeg$ carry an explicit zero case so that a group with no speeches of one polarity still yields a count, and $NegFrac$ reports the negative share in per-mille to stay within integer arithmetic.

GdprRe. This is a 42-term alternation over the regulation's official names and abbreviations in the EU's official languages, e.g. \bGDPR\b, \bDSGVO\b, \bDS-GVO\b, \bRGPD\b, \bAVG\b, \bRODO\b, Datenschutzgrundverordnung, r glement g n ral de protection des donn es, and equivalents in all 24 official EU languages.

P_{rel} (relevance).

Does the following speech mention the GDPR (General Data Protection Regulation), identified by one of its official names or abbreviations: GDPR (English), DS-GVO or Datenschutz-Grundverordnung (German), RGD (French / Spanish / Italian / Portuguese), AVG (Dutch), RODO (Polish), or equivalent official abbreviations in other EU languages? Answer 1 ONLY if the speech explicitly names or abbreviates this specific regulation. Answer 0 if: the speech does not mention the GDPR explicitly. <speech>{text}</speech>

P_{pos} (supportive stance).

In the following speech, the speaker's OVERALL or NET attitude towards the GDPR (General Data Protection Regulation / DS-GVO / RGD / AVG / RODO) is POSITIVE or SUPPORTIVE. This includes: praising the regulation as a landmark achievement or a model for the world; welcoming its adoption, entry into force, or enforcement; calling for stronger or broader application; defending it against weakening attempts; framing it as a necessary safeguard for citizens' rights; expressing satisfaction that it was agreed or achieved; or endorsing its principles even while acknowledging some complexity or room for improvement. The speaker need not use the word 'GDPR' explicitly — references to the underlying regulation by its full name or national abbreviation count. IMPORTANT: focus only on what the speaker says *about the GDPR itself*, not the overall topic of the speech. A speech primarily about terrorism, migration, or AI that merely cites 'we must respect the GDPR' expresses no positive stance about the GDPR. Answer 0 only if the speech is predominantly critical, purely procedural, or takes no stance. <speech>{text}</speech>

P_{neg} (critical stance).

In the following speech, the speaker's OVERALL or NET attitude towards the GDPR (General Data Protection Regulation / DS-GVO / RGPD / AVG / RODO) is NEGATIVE, CRITICAL, or SCEPTICAL. This includes: criticising the regulation as overly burdensome, complex, costly, or harmful; calling for its simplification, derogation, reform, or repeal; framing it as an obstacle to innovation, research, business, public health, or security; opposing specific enforcement actions as disproportionate; arguing it hinders cross-border data flows, AI development, or economic competitiveness; or expressing frustration or disappointment with how it has been applied or interpreted. The speaker may acknowledge some merits of the GDPR while still being predominantly critical. The speaker need not use the word 'GDPR' explicitly — references to the underlying regulation by its full name or national abbreviation count. IMPORTANT: focus only on what the speaker says *about the GDPR itself*, not the overall topic of the speech. A speech primarily about terrorism, migration, or security that merely says 'measures must respect the GDPR' expresses no negative stance about the GDPR. Answer 0 if: the speech is predominantly supportive of the GDPR; is purely procedural; takes no stance; or mentions the GDPR only as a legal safeguard, constraint, or compliance requirement to be respected — without actually critiquing it.

<speech>{text}</speech>

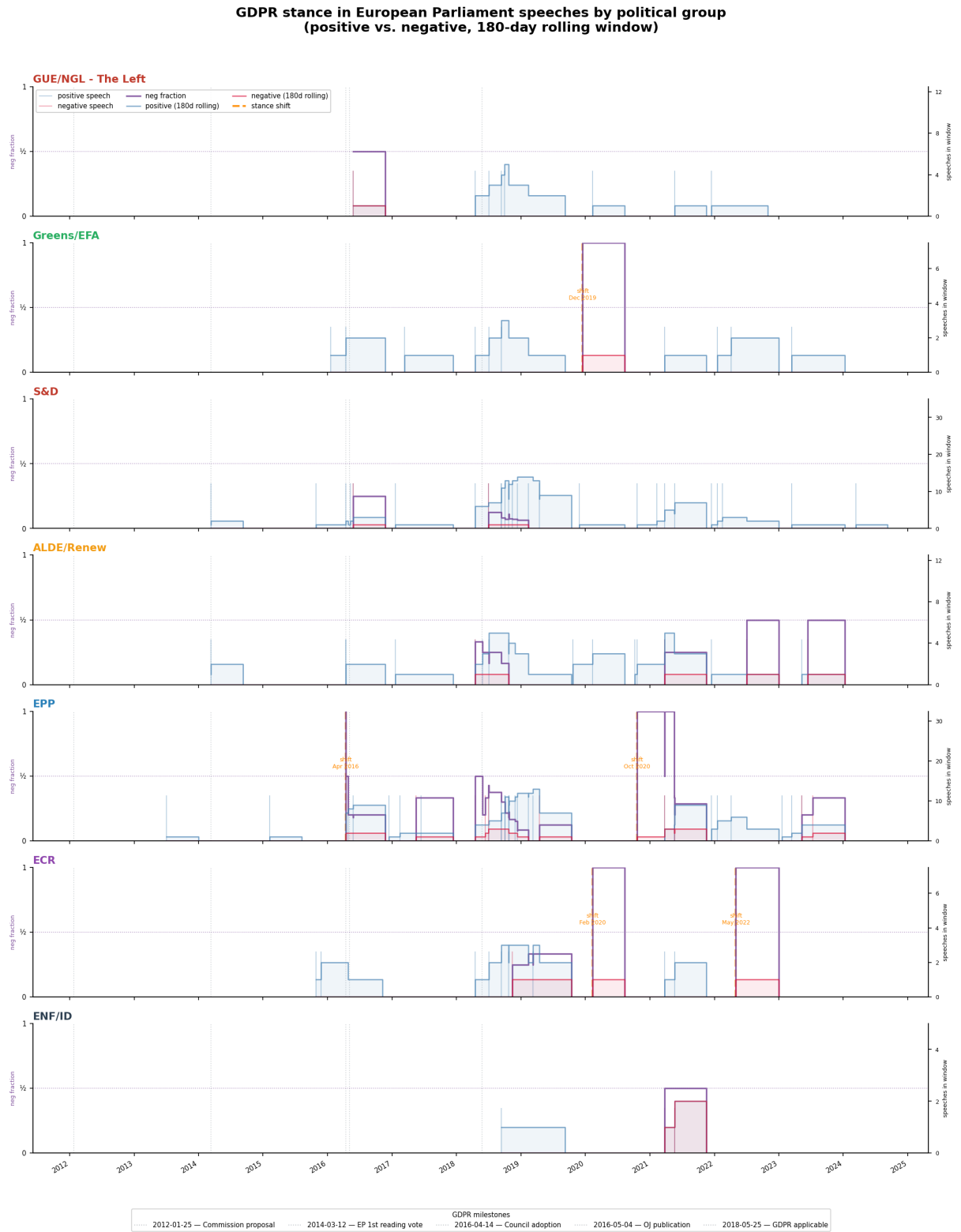


Figure 5: GDPR stance over time by political group, as returned by the case-study query. Rug marks are individual supportive (blue) and critical (red) speeches; the shaded lines are the 180-day rolling counts computed by the monitor; the purple line is the resulting negative share with its one-half threshold; dashed orange lines mark detected stance shifts. Dotted grey lines are GDPR milestones. All series are read directly off the monitor's output, so every point is traceable to the speeches that produced it.