REGREACT: Self-Correcting Multi-Agent Pipelines for Structured Regulatory Information Extraction

Anonymous ACL submission

Abstract

Extracting structured, machine-readable compliance criteria from regulatory documents remains an open challenge. Single-pass language models hallucinate structural elements, lose hierarchical relationships, and fail to resolve inter-document dependencies. We introduce REGREACT, a self-correcting multi-agent framework that decomposes regulatory information extraction into seven specialized stages, each with an *Observe–Diagnose–Repair* (ODR) loop that validates outputs against the source, correcting not only model hallucinations but also cross-reference errors in the regulations themselves. To ensure structural accuracy, REGREACT constructs a typed criterion graph; to ensure completeness, it resolves external dependencies by retrieving, summarizing, and embedding referenced legal content inline, producing self-contained outputs. Applying REGREACT to three EU Taxonomy Delegated Acts, we construct a dataset comprising 242 activities with over 4,800 hierarchical criteria, thresholds, and enriched source summaries. Evaluation against a GPT-4o single-pass baseline confirms that REGREACT outperforms it across all structural and semantic metrics[1].

1 Introduction

Regulatory frameworks governing environmental, financial, and social conduct increasingly demand automated systems that can interpret complex legal texts and extract structured compliance requirements (Zhong et al., 2020; Katz et al., 2024). The EU Taxonomy Regulation (European Parliament and Council of the European Union, 2020) exemplifies this challenge: its three Delegated Acts (European Commission, 2021, 2022, 2023) define technical screening criteria for hundreds of economic activities across six environmental objectives. These criteria are expressed in deeply nested natural language combining quantitative thresholds, implicit hierarchical groupings, cross-references to external regulations, and evaluation logic that requires understanding logical relationships among sub-criteria (see Appendix B for an example).

Despite strong advances in legal reasoning (Katz et al., 2024; Blair-Stanek et al., 2023) and LLM-based extraction (Josifoski et al., 2022; Wadhwa et al., 2023), applying language models to regulatory structuring exposes four key limitations: (i) *structural hallucination*, where a single LLM call cannot reliably maintain hierarchical relationships across dozens of criteria; (ii) *cross-stage inconsistency*, where early errors propagate uncorrected through later stages; (iii) *structural violation*, where LLM reasoning alone cannot guarantee global properties such as hierarchy well-formedness or cycle-free dependencies; and (iv) *reference incompleteness*, where criteria reference external legal acts, internal sections, and footnotes, leaving the output dependent on dozens of additional documents. No existing system produces a *self-contained* output that resolves all such dependencies inline.

We present REGREACT (**Reg**ulatory **Re**asoning with **A**gentic **C**orrec**t**ion), a multi-agent framework that addresses these challenges through three mechanisms: (1) a seven-stage pipeline with specialized agents, each equipped with an *Observe–Diagnose–Repair* (ODR) self-correction loop grounded in source document evidence, and a shared semantic memory for cross-stage consistency (§3.4, §3.2); (2) a typed criterion graph encoding six relationship types that enforces structural constraints such as hierarchy well-formedness and cycle-free inheritance (§3.5); and (3) criterion-conditioned RAG that resolves every external reference, internal cross-reference, and footnote dependency inline, producing fully *self-contained* outputs (§3.6).

Applying REGREACT to the three EU Taxonomy Delegated Acts, we construct EU-TAXOSTRUCT dataset, covering all 242 activities

[1] Code and data will be made publicly available.

Table 1: Comparison of regulatory extraction systems on key problem dimensions. ✓ = full support, ∼ = partial, ✗ = not addressed.

Feature	XTRAREG (Abualhaija et al., 2025)	Galli et al. (Galli et al., 2026)	GraphCompl. (Chung et al., 2025)	AgenticIE (Colakoglu et al., 2025)	P2T (Datla et al., 2025)	LAM (Cheng et al., 2025)	RegReAct
Task scope							
Primary regulatory text	✓	✓	✓	∼	✓	∼	✓
Structured output	✓	✓	✓	✓	✓	∼	✓
Extraction approach							
Multi-stage decomposition	✗	✓	✓	✓	✓	✓	✓
Iterative correction	✗	✗	✗	✓	✓	✓	✓
Output properties							
Hierarchical nesting	✗	∼	∼	✗	✗	✗	✓
Cross-reference handling	∼	∼	✓	✗	✗	∼	✓
Self-contained output	✗	✗	✗	✗	✗	✗	✓

with hierarchical criteria, thresholds, dependencies, and resolved source summaries. As Table 1 shows, RegReAct is the only system combining multi-agent extraction, self-correction, graph validation, and self-contained output.

Contributions. **(1)** A self-correcting multi-agent pipeline with source-grounded ODR loops (§3.4). **(2)** Criterion-conditioned RAG that produces self-contained outputs requiring no additional document consultation (§3.6). **(3)** EU-TaxoStruct, a structured dataset covering 242 EU Taxonomy activities with hierarchical criteria, thresholds, dependencies, and enriched source summaries (§4).

2 Related Work

We position RegReAct relative to prior work in Table 1; details are in Appendix A.

Legal NLP and Regulatory Extraction. Despite advances from domain-specific pre-training (Chalkidis et al., 2020), multi-task benchmarks (Chalkidis et al., 2022; Guha et al., 2023), and LLM-based reasoning (Katz et al., 2024), most work targets passage-level tasks (Hendrycks et al., 2021), leaving hierarchical compliance extraction largely unexplored. Existing approaches each cover only part of the problem: XTRAREG (Abualhaija et al., 2025) and Galli et al. (2026) extract regulatory requirements but produce flat or shallow structures without deep hierarchical nesting; GraphCompliance (Chung et al., 2025) uses graph structure for cross-references but targets compliance checking rather than extraction; and AgenticIE (Colakoglu et al., 2025) and P2T (Datla et al., 2025) incorporate correction loops but lack deep hierarchical output (Appendix A). Schmoll and Jatowt (2025) further show that LLMs achieve only moderate success on EU Taxonomy activity identification, motivating structured multi-agent approaches.

Self-Correction and RAG. Self-Refine (Madaan et al., 2023) and Reflexion (Shinn et al., 2023) established iterative refinement via self-generated feedback, but Huang et al. (2023) showed that LLMs struggle to self-correct without external signals, confirmed by CRITIC (Gou et al., 2023). Our ODR mechanism extends this principle to regulatory extraction using source-document comparison and domain-specific issue taxonomies. On the retrieval side, standard RAG (Lewis et al., 2020; Gao et al., 2023) retrieves context for generation and then discards it; adaptive strategies trigger re-retrieval when confidence drops (Jiang et al., 2023; Asai et al., 2023; Yan et al., 2024). Our criterion-conditioned RAG differs by treating each criterion as an independent retrieval query, producing criterion-focused summaries, and persisting the resolved content inline as a permanent part of the output.

3 Methodology

3.1 Problem Formulation

Let $\mathcal{D}$ be a regulatory document comprising activities $\mathcal{A} = \{a_1, \ldots, a_n\}$, where each activity a_i is represented as a semi-structured HTML fragment h_i. Our objective is to extract, for each activity, a structured representation $\mathcal{S}_i = (\mathcal{C}_i, \phi_i, \mathcal{T}_i, \mathcal{R}_i, \mathcal{G}_i, \xi_i)$ that encodes six components: hierarchical criteria $\mathcal{C}_i$ with evaluation logic ϕ_i, quantitative and temporal thresholds $\mathcal{T}_i$, external references and inter-criteria dependencies $\mathcal{R}_i$, a typed criterion graph $\mathcal{G}_i$ encoding structural relationships, and criterion-focused summaries ξ_i that resolve every reference inline. This ensures the *self-containedness property*: all information needed for compliance assessment is embedded within $\mathcal{S}_i$.

This task is challenging for several reasons: **(i)** implicit hierarchical nesting signaled by formatting rather than explicit markup; **(ii)** *unnumbered paragraphs* that must be semantically anchored to the correct parent; **(iii)** thresholds stated in paragraphs distant from the criteria they constrain; **(iv)** evaluation logic expressed implicitly through natural language connectives ("all of the following" vs. "one of the following pathways"); and **(v)** heterogeneous citation formats requiring normalization to machine-readable identifiers.

3.2 Framework Overview

REGREACT addresses these challenges through a seven-stage pipeline architecture (Figure 1) governed by five core principles designed to ensure reliable, structured extraction.

Decomposition: rather than prompting a single LLM to produce the output in one pass, we decompose extraction into stages aligned with distinct linguistic and structural competencies (§3.3). **Self-correction:** each stage implements an ODR loop (§3.4) that detects and repairs errors by comparing output against source evidence. **Structural enforcement:** a typed criterion graph (§3.5) catches structural violations that LLM reasoning alone cannot reliably detect. **Self-containedness:** every external reference, cross-reference, and footnote dependency is resolved inline through criterion-conditioned RAG (§3.6). **Cross-stage consistency:** a *shared semantic memory* maintains three registries (thresholds, cross-reference mappings, and activity metadata) populated incrementally and included in subsequent prompts.

3.3 Specialized Agent Pipeline

The pipeline comprises seven stages—structural parsing, threshold extraction, content classification, reference extraction, dependency resolution, footnote processing, and schema assembly—designed to mirror the cognitive process a regulatory expert follows when manually structuring compliance criteria (see Table 6 in Appendix E for a summary).

Stage 1: Structural Parsing. The first agent parses the regulation's HTML and recovers the logical criterion hierarchy. Although the markup contains ordered lists, its nesting does not reliably reflect the actual structure, which is conveyed through typographic cues such as indentation and numbering patterns. For instance, two mutually exclusive compliance pathways may share a single `<ol>` element with no markup distinguishing them, so the agent must infer parent–child relationships from these cues, producing structured identifiers (e.g., `SC.1.a.ii`).

A further challenge is *unnumbered paragraphs*: regulatory passages carrying substantive compliance content—verification obligations, methodology prescriptions, or background conditions—yet lacking any identifier linking them to the criteria they govern. Since identical phrasing can serve different functions depending on position and context, the agent uses semantic analysis to anchor each paragraph to the appropriate criterion, place it at the correct level in the hierarchy, assign it one of six descriptive types (Verification, Methodology, Commitment, Assessment, Replacement, or BackgroundInformation), and generate a stable identifier such as `1(f).Verification` (see Appendix E.1).

Stage 2: Threshold Extraction. Regulatory criteria often embed quantitative limits and temporal deadlines in diverse surface forms (e.g., "not exceeding 100 g CO_2e/kWh", "by 31 December 2030") that may appear far from the criterion they constrain or apply to multiple criteria through inheritance. The agent extracts and normalizes them into structured representations, verifying each against the source text to reject hallucinated quantities, and registers results in the shared semantic memory (§3.2) for cross-stage consistency.

Stage 3: Classification and Logic Inference. Beyond identifying criteria, compliance assessment requires determining their types and how they combine. The agent classifies each criterion along two dimensions—*category* (Quantitative vs. Qualitative) and *applicability* (Mandatory vs. Conditional)—and annotates it with the *evaluation logic* $\phi(c)$ that governs how its children combine. Category assignment is grounded in Stage 2 output: a criterion is Quantitative iff it carries quantitative threshold data. Applicability is inferred from the criterion's text and hierarchical position: criteria scoped by conditional phrases (e.g., "Where…", "If…") are marked Conditional, while criteria that apply unconditionally are marked Mandatory.

The evaluation logic $\phi(c)$ (AND, OR, or N_OF_K) determines whether all, any, or a specific count of a criterion's children must be satisfied. This is particularly challenging when the text is misleading: a criterion may state "the activity meets either of the following" and list children (a) through (f), suggesting a simple choice among six options, when in reality (a–e) form one joint pathway and (f) defines a separate pathway with its own sub-criteria (see Appendix B). When such alternatives are detected, a *semantic pathway detection* pass identifies which children belong to each group, producing GROUP_MEMBER edges in the criterion graph.

Stage 4: Reference Extraction. Regulatory criteria frequently delegate requirements to external legal instruments ("in accordance with Directive 2010/75/EU"), international standards ("ISO 14064-1"), or internal cross-references ("Section

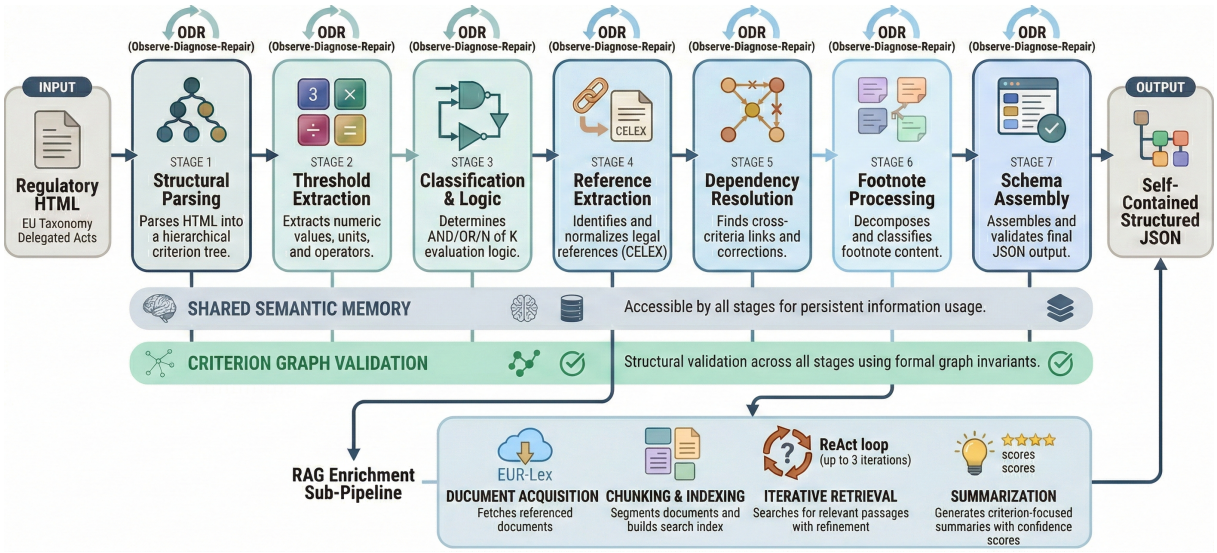

Figure 1: Overview of the REGREACT pipeline. Each stage is executed by a specialized agent with an ODR self-correction loop. The shared semantic memory maintains cross-stage consistency, the criterion graph enforces structural invariants, and the RAG sub-pipeline resolves external references inline to produce self-contained output.

5.11 of this Annex"). Not all references carry equal weight: some define operative requirements the criterion depends on, while others serve only as background context. The agent classifies each reference accordingly as `must_be_fetched`—when the criterion cannot be assessed without consulting the source—or `citation_only`—when it provides supplementary context only. This classification determines which references enter the retrieval pipeline (§3.6).

For EU legislation, references are normalized to machine-readable CELEX identifiers (e.g., "Regulation (EU) 2018/1999" → `32018R1999`) through multi-step parsing and validation (see Appendix E.2). Each identifier is then verified against EUR-Lex to confirm the referenced document exists. If verification fails, the pipeline enters a ReAct-style correction loop: it observes the error, reasons about its likely cause—such as a regulation misclassified as a directive or a malformed year—applies a targeted fix, and re-verifies. This repeats for up to three attempts; if all fail, the pipeline falls back to a title-based EUR-Lex search. References that remain unresolved are flagged for manual review.

Stage 5: Dependency Resolution. Criteria in regulatory documents rarely stand alone: one criterion may apply only if another is satisfied, and some thresholds are defined in one clause yet silently referenced by another. Since these relationships are never explicitly marked in the HTML, the agent infers them from the text and organizes them into three types. *Conditional dependencies* arise when a criterion activates only under specific conditions—for example, a verification obligation that applies only when the transitional compliance pathway is selected ("if the facility complies via point 1(f)…"). *Threshold inheritance* occurs when a criterion points to a value defined elsewhere (e.g., "the emissions referred to in point 1(f)(a)"); the agent copies the referenced threshold and tags it with a `[THRESHOLD_FROM:1(f)(a)]` annotation. *Cross-reference corrections* handle a less obvious problem: errors in the regulation itself, where a criterion is cited under the wrong identifier—for instance, "point 1(b)" when the actual target is criterion 1(f). The agent detects such mismatches through semantic analysis and records every correction with full provenance (e.g., `[CORR FROM:1(b) TO:1(f)]`, see Appendix B). All three relationship types are encoded as typed edges in the criterion graph (§3.5).

Stage 6: Footnote Processing. Footnotes in regulatory documents often contain critical compliance details—legal references, technical definitions, official standards, and explanatory notes—embedded as hidden text within the HTML. A single footnote may combine multiple distinct items of different types: for instance, footnote 230 in the Appendix example contains 13 references mixing EU legal acts with international standards, each requiring independent classification as `must_be_fetched` or `citation_only`. The agent extracts each foot-

note's content, decomposes it into individual items, categorizes them, and links each footnote to the criterion whose text contains its marker.

Stage 7: Schema Assembly. The final agent assembles all extracted components into a validated JSON schema, performing format normalization and cross-field consistency checks. These checks verify that category labels agree with threshold data, that evaluation logic matches child count, and that complex objects with no meaningful content are nullified.

3.4 Self-Correction via ODR

Each agent's output is iteratively refined through a self-correction loop. Building on ReAct (Yao et al., 2022) and iterative self-refinement (Madaan et al., 2023; Shinn et al., 2023), ODR adapts these ideas to structured information extraction: rather than relying solely on the model's self-assessment, the agent compares its output against the source document to ground error detection in verifiable evidence.

Observe–Diagnose–Repair. In the **observe** step, the agent compares its structured output against the original HTML and enumerates discrepancies categorized by type and severity (see Appendix C for the full issue taxonomy). This provides the external feedback signal that Huang et al. (2023) identify as necessary for effective self-correction. In the **diagnose** step, the LLM analyzes observed issues and previous attempts to identify root causes (see Appendix D.14). In the **repair** step, the diagnosis constructs a targeted correction prompt, and the agent re-extracts with this guidance.

The loop terminates when confidence $\geq \tau$ with no critical issues, when the same issues repeat, or upon timeout, returning the highest-confidence checkpoint. When the loop exhausts its iterations without resolving critical issues, the output is flagged for human review rather than silently accepted. We use $\tau = 0.7$ and $k_{\max} = 3$ throughout our experiments.

3.5 Graph-Based Structural Validation

Since each agent operates independently, their outputs may contradict each other. For example, the evaluation logic agent (Stage 3) may determine that a criterion requires all three of its children to be satisfied, while the structural parser (Stage 1) actually extracted four children for that criterion. Similarly, a dependency identified in Stage 5 may point to a criterion that was never extracted. To catch such inconsistencies, we unify all stage outputs into a single criterion graph $\mathcal{G}_i = (V_i, E_i)$ per activity, where vertices represent criteria and edges encode six relationship types: HIERARCHY (parent–child nesting), GROUP_MEMBER (sibling grouping under shared evaluation logic), INHERITS_THRESHOLD (threshold propagation between criteria), REFERENCES (citation of external sources), DEPENDS_ON (conditional inter-criteria dependencies), and CORRECTS (cross-reference corrections identified by semantic analysis).

Construction and Validation. As each stage completes, its relationships are added as typed edges—hierarchy from Stage 1, grouping and inheritance from Stages 2–3, references and dependencies from Stages 4–5, and corrections from Stage 6—with per-edge checks ensuring that the target criterion exists. The graph also repositions criteria that the flat HTML caused to be mis-nested, using their identifiers to restore the correct hierarchy, and links unnumbered paragraphs to the correct parent. Once fully constructed, a global validation pass checks consistency at two levels: *structurally*, verifying that hierarchy edges form a cycle-free tree, that the evaluation logic of each criterion matches its actual child count, and that no criteria are left disconnected; and at the *regulatory* level, enforcing task-specific constraints including an *Evaluation Participation Rule* that excludes ancillary nodes (e.g., background information) from child counts so that non-binding content does not distort the compliance logic inferred in Stage 3.

3.6 Criterion-Conditioned RAG

Regulatory criteria frequently refer to external legal instruments (e.g., "in accordance with Directive 2010/75/EU") and to other sections of the same regulation (e.g., "Section 5.11 of this Annex") without restating the referenced content. To make the output self-contained, the RAG module retrieves and embeds the relevant content directly within each criterion through a four-stage pipeline, operating on references from Stage 4 and footnote items from Stage 6 that carry CELEX identifiers.

Document Acquisition. Referenced documents are identified by their CELEX identifiers and fetched in PDF format from EUR-Lex, then converted to structured Markdown using MinerU (Wang et al., 2024).

Semantic Chunking and Indexing. Documents are chunked using a structure-aware strategy that respects article and paragraph boundaries. Chunks are encoded with a ColBERT late-interaction model (Khattab and Zaharia, 2020) and indexed using Hierarchical Navigable Small World (HNSW) approximate nearest-neighbour search (Malkov and Yashunin, 2018) on a per-document basis.

Iterative Criterion-Conditioned Retrieval. Rather than issuing a generic query, the LLM rewrites each criterion into a retrieval-optimized natural-language question enriched with the activity name, environmental objective, and article reference (e.g., *"What emission threshold does Article 29(4)(a) establish for electricity generation using solar photovoltaic technology?"*). Each retrieval iteration fuses ColBERT dense scores with BM25 sparse scores via reciprocal rank fusion (RRF; Cormack et al. 2009) and reranks candidates using ColBERT MaxSim scoring. Retrieval follows an iterative refinement loop inspired by ReAct (Yao et al., 2022): the LLM evaluates retrieved passages against the criterion and identifies information gaps (e.g., "threshold value not found", "temporal constraint missing"). If confidence remains below a threshold, the gap description generates a refined query, and retrieval is repeated for up to three iterations.

Query-Focused Summarization. The top-ranked chunks are passed to the LLM together with the original criterion, so that the generated summary retains only the information relevant to that specific compliance requirement— thresholds, definitions, exceptions, and conditions. To preserve factual accuracy, the summarizer enforces a verbatim quote limit, ensuring that summaries are genuine distillations rather than reproductions. The resolved content is persisted inline, eliminating the need to consult the original reference documents.

4 The EU-TAXOSTRUCT Dataset

We construct EU-TAXOSTRUCT by applying REGREACT to the three EU Taxonomy Delegated Acts (European Commission, 2021, 2022, 2023), a domain that concentrates all four challenges identified in §1. The resulting dataset covers 242 activities with over 4,800 hierarchical criterion nodes across six environmental objectives, each encoding evaluation logic, thresholds, inter-criteria dependencies, and resolved reference summaries inline (output schema in Appendix F).

Gold Annotation. To evaluate extraction quality, we constructed a gold-annotated subset of 100 activities via stratified sampling across three structural complexity tiers and all three Delegated Acts. Two PhD students served as annotators, each independently reviewing 50 activities across nine evaluation dimensions following the extraction specification (Appendix F). Inter-annotator agreement, measured on a shared subset of 15 activities, yielded Cohen's $\kappa = 0.84$–0.91 across all evaluated dimensions (Appendix G).

5 Experimental Setup

Evaluation Strategy. Direct comparison with related work (Table 1) is infeasible because each operates on different regulatory texts with incompatible output schemas. We instead evaluate on n=100 manually annotated activities and compare against a *GPT-4o single-pass* baseline that receives the same schema and prompt in one call, testing whether a larger model alone can match multi-agent decomposition. We evaluate along three axes: structural and classification accuracy, semantic equivalence via an LLM judge, and RAG summary quality.

Structural & Classification Metrics (n=100). Four metrics are computed against gold annotations. **Structural F1** aligns predicted and gold criterion trees by `criterion_id` matching, scoring each pair in $[0, 1]$ based on placement correctness and schema completeness (Appendix E.3). **Category Accuracy** measures correct Quantitative/Qualitative assignment. **Applicability Accuracy** measures correct Mandatory/Conditional assignment, which distinguishes universally applicable criteria from those gated by conditions. **Evaluation Logic Accuracy** checks whether the predicted logic (AND, OR, N_OF_K, LEAF) matches gold, testing the ability to distinguish conjunctive from disjunctive requirements.

Semantic Equivalence Metrics ($n \leq 100$). A GPT-4o judge scores four dimensions on a 0–5 scale (0 = both sides empty; 1–5 from wrong to full equivalence), receiving the source text, gold field, and extracted field (rubrics in Appendix D.19). Scores are averaged per activity over non-zero nodes; n varies across dimensions. **Threshold** checks value, unit, operator, and temporal constraints. **Reference**

Table 2: Evaluation results for REGREACT on n=100 gold-annotated activities. Structural and classification metrics are percentages; semantic equivalence uses a 1–5 GPT-4o judge scale (§5). Activities with all-zero scores are excluded, so n varies.

Metric	Mean	Std	Median	n
Structural & Classification (%)				
Structural F1	94.12	3.56	95.2	100
Category Accuracy	98.6	2.62	100.0	100
Applicability Accuracy	97.24	3.16	98.4	100
Evaluation Logic Accuracy	93.4	3.60	94.0	100
Semantic Equivalence (1–5)				
Threshold	4.43	0.54	5.00	74
Reference	4.77	0.22	5.00	100
Footnote	4.48	0.60	4.50	99
Dependency	4.63	0.80	5.00	99

Table 3: REGREACT vs. GPT-4o single-pass baseline (n=100 gold-annotated activities). Structural and classification metrics are percentages; semantic equivalence as mean Likert scores (1–5).

Metric	REGREACT	GPT-4o	Δ
Structural & Classification (%)			
Structural F1	94.12	78.6	+15.52
Category Accuracy	98.6	90.2	+8.4
Applicability Accuracy	97.24	85.7	+11.54
Evaluation Logic Accuracy	93.4	80.3	+13.1
Semantic Equivalence (1–5)			
Threshold	4.43	3.23	+1.20
Reference	4.77	3.34	+1.43
Footnote	4.48	3.12	+1.36
Dependency	4.63	2.96	+1.67

verifies completeness and correctness of legal citations, including CELEX normalization and `must_be_fetched`/`citation_only` classification. **Footnote** checks faithful capture and correct criterion linkage. **Dependency** assesses conditional gating, threshold inheritance, and cross-reference corrections against the gold annotation.

RAG Summary Quality. Each generated summary is scored by a GPT-4o judge on a 1–5 Likert scale (prompts in Appendix D.20) along four dimensions. **Faithfulness** measures the proportion of summary claims grounded in retrieved passages, penalizing hallucination but not omission. **Relevance** classifies claims into three tiers (directly relevant, contextual, unrelated) relative to the criterion. **Completeness** evaluates whether compliance-critical information is captured, weighted by element importance (major vs. minor). **Coverage** measures whether the summary addresses the criterion's information needs, weighted by need priority (primary vs. secondary).

6 Results and Analysis

6.1 Extraction Quality

Structural & Classification Accuracy. On the 100 gold-annotated activities (Table 2), REGREACT achieves 94.12% *structural F1*, reconstructing criterion trees with high fidelity; residual errors concentrate in activities with deeply nested alternative pathways. *Category accuracy* is highest at 98.6%, as the deterministic category rule in Stage 3 leaves little room for error once thresholds have been correctly extracted. *Applicability accuracy* reaches 97.24%; residual errors arise when conditional language is ambiguous between a compliance requirement and an applicability condition. *Evaluation logic accuracy* (93.4%) reflects the difficulty of distinguishing conjunctive from disjunctive requirements in ambiguous (chapeau) text.

Semantic Equivalence. The LLM-judge scores confirm strong semantic fidelity across all four dimensions, with all means exceeding 4.4 on the 1–5 scale. Reference equivalence achieves 4.77, the highest among the four dimensions, reflecting the effectiveness of deterministic CELEX normalization. Dependency equivalence reaches 4.63, reflecting the inherent complexity of dependency chains spanning multiple criteria. Threshold equivalence scores 4.43 (n=74), with the multi-component nature of thresholds (value, unit, operator, temporal constraint) making this the most demanding semantic dimension. Footnote equivalence is 4.48, with residual errors arising from footnotes that combine multiple items of different types.

Baseline Comparison. Table 3 compares REGREACT against a GPT-4o single-pass baseline that receives identical instructions and output schema but produces everything in one call. Although the baseline relies on a larger model, REGREACT—using DeepSeek-R1-Distill-Qwen-32B (Guo et al., 2025)—outperforms it on all eight metrics, confirming that decomposition, self-correction, and graph validation matter more than model scale for this task.

The largest structural gaps appear on structural F1 (+15.52) and evaluation logic (+13.1), both of which demand hierarchical consistency across dozens of criteria—precisely where a single call loses track of parent–child relationships. On the

Table 4: RAG summary quality scores (GPT-4o judge, 1–5 Likert).

Metric	Mean	Std	Median
Faithfulness	4.61	1.13	5
Relevance	4.14	1.02	4
Completeness	4.07	0.85	4
Coverage	4.01	1.08	4

Table 5: Quality by ReAct iteration count. Ret. Conf. is the pipeline's self-assessed retrieval confidence.

Iter.	Count (%)	Faith.	Relev.	Compl.	Cover.	Ret. Conf.
1	535 (75.4%)	4.74	4.30	4.17	4.30	0.92
2	88 (12.4%)	4.18	3.80	3.94	3.33	0.86
3	86 (12.1%)	4.28	3.48	3.64	2.93	0.67

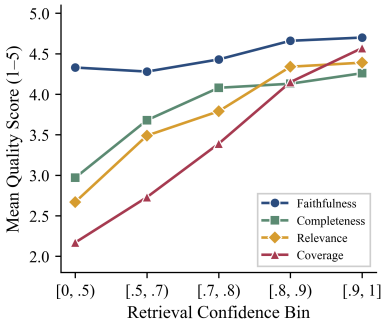

Figure 2: Mean quality scores per retrieval confidence bin, averaged over criterion–source pairs. Faithfulness remains stable while relevance, completeness, and coverage increase monotonically.

semantic side, dependency equivalence gains the most (+1.67), as detecting conditional gating and threshold inheritance requires the dedicated dependency stage and shared memory absent from the baseline. The remaining dimensions each improve by over one point, reflecting the value of specialized agents for compound thresholds, multi-item references, and footnotes.

6.2 RAG Enrichment Quality

We evaluate the RAG enrichment module on 709 criterion–source pairs spanning 242 activities and 93 EUR-Lex documents.

Faithfulness is the strongest dimension (4.61), indicating that summary claims are directly traceable to retrieved passages—critical in the regulatory domain, where hallucinated obligations could lead to incorrect compliance assessments. Relevance and completeness both exceed 4.0, confirming that summaries are targeted and capture major compliance-critical elements, while coverage confirms that retrieved passages address each criterion's core information needs.

ReAct Loop Analysis. The retrieval pipeline employs a ReAct loop (Yao et al., 2022) that reformulates queries when confidence falls below a threshold. Table 5 stratifies quality by iteration count.

Three observations emerge. First, 75.4% of pairs resolve in a single iteration with the highest quality across all metrics, indicating effective initial query formulation. Second, multi-iteration cases represent inherently harder retrieval tasks: relevance, completeness, and coverage decrease with iteration count, while the monotonic confidence decrease ($0.92 \rightarrow 0.67$) confirms that the pipeline correctly identifies difficult queries. Third, faithfulness remains stable (≥ 4.18) across all iteration counts, showing that the pipeline maintains grounding quality under difficult retrieval rather than compensating with hallucination.

Confidence Calibration. We examine whether confidence scores predict output quality, enabling automated quality filtering. Figure 2 partitions the 709 criterion–source pairs into five confidence bins and reports the mean quality score per metric.

Relevance, completeness, and coverage increase monotonically across all confidence bins. Faithfulness remains high (≥ 4.28) across all bins, varying less than the other three dimensions. The clear separation between low- and high-confidence outputs, particularly for coverage (2.17 vs. 4.57), confirms confidence scores carry meaningful quality signal.

7 Conclusion

We presented REGREACT, a self-correcting multi-agent framework that decomposes regulatory information extraction into seven specialized stages with Observe–Diagnose–Repair loops grounded in source evidence, a typed criterion graph for structural validation, and criterion-conditioned RAG for inline reference resolution, producing self-contained structured outputs. Applying REGREACT to three EU Taxonomy Delegated Acts yielded EU-TAXOSTRUCT, a dataset of 242 activities with hierarchical criteria, thresholds, and enriched source summaries. Evaluation against a GPT-4o single-pass baseline showed consistent gains across all metrics, highlighting the value of structured multi-agent pipelines over monolithic generation for complex regulatory texts. Future work will target other regulatory domains and languages.

Limitations

Three practical limitations remain: (1) references to ISO/EN standards cannot be enriched inline as their content is behind paywalls, limiting self-containedness for those references; (2) the current evaluation covers English-language regulations; multilingual EU texts remain untested; and (3) the extracted dataset reflects the regulations as published at the time of processing; subsequent amendments require re-execution of the pipeline.

References

Sallam Abualhaija, Marcello Ceci, Nicolas Sannier, Domenico Bianculli, Salomé Lannier, Martina Siclari, Olivier Voordeckers, and Stanisław Tosza. 2025. Llm-assisted extraction of regulatory requirements: A case study on the gdpr. In *2025 IEEE 33rd International Requirements Engineering Conference (RE)*, pages 142–154. IEEE.

Mohammed Ali, Abdelrahman Abdallah, Amit Agarwal, Hitesh Laxmichand Patel, and Adam Jatowt. 2026. Recor: Reasoning-focused multi-turn conversational retrieval benchmark. *arXiv preprint arXiv:2601.05461*.

Mohammed Ali, Abdelrahman Abdallah, and Adam Jatowt. 2025. Sustainableqa: A comprehensive question answering dataset for corporate sustainability and eu taxonomy reporting. *arXiv preprint arXiv:2508.03000*.

Akari Asai, Zeqiu Wu, Yizhong Wang, Avirup Sil, and Hannaneh Hajishirzi. 2023. Self-rag: Learning to retrieve, generate, and critique through self-reflection. In *The Twelfth International Conference on Learning Representations*.

Andrew Blair-Stanek, Nils Holzenberger, and Benjamin Van Durme. 2023. Can gpt-3 perform statutory reasoning? In *Proceedings of the Nineteenth International Conference on Artificial Intelligence and Law*, pages 22–31.

Ilias Chalkidis, Manos Fergadiotis, Prodromos Malakasiotis, Nikolaos Aletras, and Ion Androutsopoulos. 2020. Legal-bert: The muppets straight out of law school. In *Findings of the association for computational linguistics: EMNLP 2020*, pages 2898–2904.

Ilias Chalkidis, Abhik Jana, Dirk Hartung, Michael Bommarito, Ion Androutsopoulos, Daniel Katz, and Nikolaos Aletras. 2022. Lexglue: A benchmark dataset for legal language understanding in english. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4310–4330.

Linze Chen, Yufan Cai, Zhe Hou, and Jinsong Dong. 2025. Towards trustworthy legal ai through llm agents and formal reasoning. *arXiv preprint arXiv:2511.21033*.

Weize Chen, Yusheng Su, Jingwei Zuo, Cheng Yang, Chenfei Yuan, Chi-Min Chan, Heyang Yu, Yaxi Lu, Yi-Hsin Hung, Chen Qian, and 1 others. 2023. Agentverse: Facilitating multi-agent collaboration and exploring emergent behaviors. In *The Twelfth International Conference on Learning Representations*.

Jiseong Chung, Ronny Ko, Wonchul Yoo, Makoto Onizuka, Sungmok Kim, Tae-Wan Kim, and Won-Yong Shin. 2025. Graphcompliance: Aligning policy and context graphs for llm-based regulatory compliance. *arXiv preprint arXiv:2510.26309*.

Gaye Colakoglu, Gürkan Solmaz, and Jonathan Fürst. 2025. Agenticie: An adaptive agent for information extraction from complex regulatory documents. *arXiv preprint arXiv:2509.11773*.

Gordon V Cormack, Charles LA Clarke, and Stefan Buettcher. 2009. Reciprocal rank fusion outperforms condorcet and individual rank learning methods. In *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, pages 758–759.

Gautam Varma Datla, Anudeep Vurity, Tejaswani Dash, Tazeem Ahmad, Mohd Adnan, and Saima Rafi. 2025. Executable governance for ai: Translating policies into rules using llms. *arXiv preprint arXiv:2512.04408*.

Darren Edge, Ha Trinh, Newman Cheng, Joshua Bradley, Alex Chao, Apurva Mody, Steven Truitt, Dasha Metropolitansky, Robert Osazuwa Ness, and Jonathan Larson. 2024. From local to global: A graph rag approach to query-focused summarization. *arXiv preprint arXiv:2404.16130*.

European Commission. 2021. Commission delegated regulation (EU) 2021/2139 supplementing Regulation (EU) 2020/852 by establishing the technical screening criteria. Official Journal of the European Union, L 442, 9.12.2021, p. 1.

European Commission. 2022. Commission delegated regulation (EU) 2022/1214 amending delegated regulation (EU) 2021/2139 as regards economic activities in certain energy sectors. Official Journal of the European Union, L 188, 15.7.2022, p. 1. Complementary Delegated Act covering nuclear energy and fossil gas activities.

European Commission. 2023. Commission delegated regulation (EU) 2023/2486 supplementing Regulation (EU) 2020/852 by establishing the technical screening criteria for the remaining four environmental objectives. Official Journal of the European Union, OJ L, 2023/2486, 21.11.2023.

European Parliament and Council of the European Union. 2020. Regulation (EU) 2020/852 of the European Parliament and of the Council on the establishment of a framework to facilitate sustainable invest-

ment. Official Journal of the European Union, L 198, 22.6.2020, p. 13–43.

Federico Galli, Thiago Raulino Dal Pont, Galileo Sartor, and Giuseppe Contissa. 2026. Approaching the ai act... with ai: Llms and knowledge graphs to extract and analyse obligations. *Computer Law & Security Review*, 60:106230.

Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yixin Dai, Jiawei Sun, Haofen Wang, Haofen Wang, and 1 others. 2023. Retrieval-augmented generation for large language models: A survey. *arXiv preprint arXiv:2312.10997*, 2(1):32.

Zhibin Gou, Zhihong Shao, Yeyun Gong, Yelong Shen, Yujiu Yang, Nan Duan, and Weizhu Chen. 2023. Critic: Large language models can self-correct with tool-interactive critiquing. *arXiv preprint arXiv:2305.11738*.

Neel Guha, Julian Nyarko, Daniel Ho, Christopher Ré, Adam Chilton, Alex Chohlas-Wood, Austin Peters, Brandon Waldon, Daniel Rockmore, Diego Zambrano, and 1 others. 2023. Legalbench: A collaboratively built benchmark for measuring legal reasoning in large language models. *Advances in neural information processing systems*, 36:44123–44279.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, and 1 others. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*.

Dan Hendrycks, Collin Burns, Anya Chen, and Spencer Ball. 2021. Cuad: An expert-annotated nlp dataset for legal contract review. *arXiv preprint arXiv:2103.06268*.

Sirui Hong, Mingchen Zhuge, Jonathan Chen, Xiawu Zheng, Yuheng Cheng, Jinlin Wang, Ceyao Zhang, Zili Wang, Steven Ka Shing Yau, Zijuan Lin, and 1 others. 2023. Metagpt: Meta programming for a multi-agent collaborative framework. In *The twelfth international conference on learning representations*.

Jie Huang, Xinyun Chen, Swaroop Mishra, Huaixiu Steven Zheng, Adams Wei Yu, Xinying Song, and Denny Zhou. 2023. Large language models cannot self-correct reasoning yet. *arXiv preprint arXiv:2310.01798*.

Zhengbao Jiang, Frank F Xu, Luyu Gao, Zhiqing Sun, Qian Liu, Jane Dwivedi-Yu, Yiming Yang, Jamie Callan, and Graham Neubig. 2023. Active retrieval augmented generation. In *Proceedings of the 2023 conference on empirical methods in natural language processing*, pages 7969–7992.

Martin Josifoski, Nicola De Cao, Maxime Peyrard, Fabio Petroni, and Robert West. 2022. Genie: Generative information extraction. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4626–4643.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 conference on empirical methods in natural language processing (EMNLP)*, pages 6769–6781.

Daniel Martin Katz, Michael James Bommarito, Shang Gao, and Pablo Arredondo. 2024. Gpt-4 passes the bar exam. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 382(2270).

Omar Khattab and Matei Zaharia. 2020. Colbert: Efficient and effective passage search via contextualized late interaction over bert. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval*, pages 39–48.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, and 1 others. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in neural information processing systems*, 33:9459–9474.

Junlong Li, Yiheng Xu, Lei Cui, and Furu Wei. 2022. Markuplm: Pre-training of text and markup language for visually rich document understanding. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6078–6087.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, and 1 others. 2023. Self-refine: Iterative refinement with self-feedback. *Advances in neural information processing systems*, 36:46534–46594.

Yu A Malkov and Dmitry A Yashunin. 2018. Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs. *IEEE transactions on pattern analysis and machine intelligence*, 42(4):824–836.

Joel Niklaus, Veton Matoshi, Pooja Rani, Andrea Galassi, Matthias Stürmer, and Ilias Chalkidis. 2023. Lextreme: A multi-lingual and multi-task benchmark for the legal domain. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 3016–3054.

Liangming Pan, Michael Saxon, Wenda Xu, Deepak Nathani, Xinyi Wang, and William Yang Wang. 2024. Automatically correcting large language models: Surveying the landscape of diverse automated correction strategies. *Transactions of the Association for Computational Linguistics*, 12:484–506.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th*

international joint conference on natural language processing (EMNLP-IJCNLP), pages 3982–3992.

Jonathan Schmoll and Adam Jatowt. 2025. Automated analysis of sustainability reports: Using large language models for the extraction and prediction of eu taxonomy-compliant kpis. *arXiv preprint arXiv:2512.24289*.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. 2023. Reflexion: Language agents with verbal reinforcement learning. *Advances in neural information processing systems*, 36:8634–8652.

Amin Sleimi, Nicolas Sannier, Mehrdad Sabetzadeh, Lionel Briand, and John Dann. 2018. Automated extraction of semantic legal metadata using natural language processing. In *2018 IEEE 26th International Requirements Engineering Conference (RE)*, pages 124–135. IEEE.

Somin Wadhwa, Silvio Amir, and Byron C Wallace. 2023. Revisiting relation extraction in the era of large language models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15566–15589.

Bin Wang, Chao Xu, Xiaomeng Zhao, Linke Ouyang, Fan Wu, Zhiyuan Zhao, Rui Xu, Kaiwen Liu, Yuan Qu, Fukai Shang, and 1 others. 2024. Mineru: An open-source solution for precise document content extraction. *arXiv preprint arXiv:2409.18839*.

Qifan Wang, Yi Fang, Anirudh Ravula, Fuli Feng, Xiaojun Quan, and Dongfang Liu. 2022a. Webformer: The web-page transformer for structure information extraction. In *Proceedings of the ACM Web Conference 2022*, pages 3124–3133.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2022b. Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171*.

William Watson, Nicole Cho, Nishan Srishankar, Zhen Zeng, Lucas Cecchi, Daniel Scott, Suchetha Siddagangappa, Rachneet Kaur, Tucker Balch, and Manuela Veloso. 2025. Law: Legal agentic workflows for custody and fund services contracts. In *Proceedings of the 31st International Conference on Computational Linguistics: Industry Track*, pages 583–594.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, and 1 others. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.

Sean Welleck, Ximing Lu, Peter West, Faeze Brahman, Tianxiao Shen, Daniel Khashabi, and Yejin Choi. 2022. Generating sequences by learning to self-correct. *arXiv preprint arXiv:2211.00053*.

Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Beibin Li, Erkang Zhu, Li Jiang, Xiaoyun Zhang, Shaokun Zhang, Jiale Liu, and 1 others. 2024. Autogen: Enabling next-gen llm applications via multi-agent conversations. In *First conference on language modeling*.

Shi-Qi Yan, Jia-Chen Gu, Yun Zhu, and Zhen-Hua Ling. 2024. Corrective retrieval augmented generation. *arXiv preprint arXiv:2401.15884*.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. 2023. Tree of thoughts: Deliberate problem solving with large language models. *Advances in neural information processing systems*, 36:11809–11822.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R Narasimhan, and Yuan Cao. 2022. React: Synergizing reasoning and acting in language models. In *The eleventh international conference on learning representations*.

Haoxi Zhong, Chaojun Xiao, Cunchao Tu, Tianyang Zhang, Zhiyuan Liu, and Maosong Sun. 2020. How does nlp benefit legal system: A summary of legal artificial intelligence. In *Proceedings of the 58th annual meeting of the association for computational linguistics*, pages 5218–5230.

Appendix Contents

A **Extended Related Work** 13

B **Example Extraction Output** 15

C **ODR Self-Correction Details** 27

D **Prompt Templates** 27

D.1–D.12 Pipeline Stage Prompts (Stages 1–7) 27
D.13–D.14 ODR Prompts 31
D.15–D.18 RAG Prompts 32
D.19–D.20 Evaluation Judge Prompts 32

E **Supplementary Methodology Details** 35

E.1 Semantic Anchoring Procedure 35
E.2 CELEX Identifier Parsing 36
E.3 Structural F1 Computation 36
E.4 RAG Query Refinement 37

F **Output Schema** 37

G **Gold Annotation Details** 41

H **Extended Evaluation** 41

H.1 Task Definition 41
H.2 Self-Refine Baseline 41
H.3 Computational Cost 42
H.4 Cross-Model Judge Validation 42
H.5 Cross-Reference Correction Evidence 42
H.6 Rationale for Seven Stages 42

A Extended Related Work

This appendix provides a detailed discussion of prior work organized by the four research threads that REGREACT builds upon.

A.1 Legal and Regulatory NLP

The application of NLP to the legal domain has progressed from rule-based systems to neural approaches. LEGAL-BERT (Chalkidis et al., 2020) demonstrated that domain-specific pretraining improves legal text classification, while LexGLUE (Chalkidis et al., 2022) and LEXTREME (Niklaus et al., 2023) established multi-task benchmarks across jurisdictions. LegalBench (Guha et al., 2023) expanded evaluation to 162 tasks, and CUAD (Hendrycks et al., 2021) provided expert-annotated data for contract clause identification. At the LLM level, Katz et al. (2024) showed GPT-4 passing the Bar Examination, while Blair-Stanek et al. (2023) investigated statutory reasoning limitations.

The vast majority of this work operates at the *classification* or *question-answering* level over individual passages (Zhong et al., 2020). The problem we address, extracting complete hierarchical compliance structures from regulation-length documents, remains largely unexplored. Early work on automated legal metadata extraction (Sleimi et al., 2018) targeted deontic statements but did not produce machine-readable compliance hierarchies. XTRAREG (Abualhaija et al., 2025) used LLM+RAG to extract privacy requirements from the GDPR, achieving 82% accuracy on access rights, but produces flat requirement lists without hierarchical nesting, inter-criteria dependencies, or self-contained output. In the sustainability domain, Schmoll and Jatowt (2025) evaluate LLMs on EU Taxonomy KPI extraction from 190 corporate reports, finding that models achieve only moderate success and function best as assistive tools. Complementing this, Ali et al. (2025) introduce SustainableQA, a question answering benchmark over ESG and EU Taxonomy documents, and Ali et al. (2026) present RECOR, a reasoning-intensive multi-turn conversational retrieval benchmark spanning sustainability among other domains; both works underscore the difficulty of regulatory NLP and motivate structured, reasoning-aware pipelines such as REGREACT.

A.2 LLM-Based Regulatory Extraction

Recent work has applied LLMs directly to regulatory information extraction. We organize the most relevant systems by their core contribution.

Multi-stage pipelines for EU legislation. Galli et al. (2026) present the most comprehensive pipeline for EU legislation, targeting the AI Act with a four-stage workflow: (1) identification of obligation-bearing provisions, (2) filtering of deontic statements, (3) analysis of deontic content (addressees, predicates, conditions), and (4) construction of a searchable knowledge graph. Using LLaMA 3.3 70B with five expert evaluators, they achieve 93% precision on obligation filtering and over 99% accuracy on obligation type classification, identifying 729 obligations across the AI Act. The knowledge graph captures article-to-obligation relationships, providing partial hierarchical structure; however, obligations themselves are flat triples without nested sub-criteria, and the pipeline includes no self-correction mechanism.

Graph-based compliance representations. GraphCompliance (Chung et al., 2025) constructs a Policy Graph from GDPR provisions, encoding normative structure through typed nodes and cross-reference edges that link articles to exceptions, definitions, and related provisions. A separate Context Graph formalizes runtime events as subject/action/object triples, and a deterministic Compliance Gate performs structural analysis via reference traversal and exception chaining. GraphCompliance is the closest prior work to ours in its use of graph structure for cross-reference handling; however, it targets compliance *checking* (producing compliant/non-compliant judgments) rather than criteria *extraction*, and does not implement iterative correction or self-contained output.

Agentic self-correction on regulatory documents. AgenticIE (Colakoglu et al., 2025) introduces a planner/executor/responder agent architecture for extracting key-value pairs from EU Declarations of Performance, standardized product certification forms issued under the Construction Products Regulation. When the executor fails to extract a field, a self-repair loop re-prompts with failure feedback. AgenticIE is the closest work to ours for agentic correction on EU regulatory documents; however, it targets product certification forms rather than legislative text, uses a single adap-

tive agent rather than multiple specialized agents, and produces shallow key-value structures (up to two levels of nesting for grouped product properties) without deep hierarchical criterion trees or cross-reference resolution.

P2T (Datla et al., 2025) converts AI governance policies (EU AI Act, NIST AI RMF) into executable rules through a six-stage pipeline. A mine/judge/repair loop iteratively refines extracted rules: an LLM judge scores each rule, and a repair agent addresses identified deficiencies. An SMT solver then detects logical conflicts between rules. P2T shares our philosophy of iterative correction with explicit quality assessment, but produces flat atomic rules without hierarchical nesting, cross-reference resolution, or self-contained output.

Legal agentic workflows. LAW (Watson et al., 2025) introduces agentic workflows for custody and fund services contracts, orchestrating specialized agents and domain-specific tools for structured extraction. LAW demonstrates the effectiveness of agent specialization for legal extraction, but addresses a fundamentally different task (contract analysis with known clause types) and does not implement self-correction or graph-based validation.

General-purpose multi-agent frameworks (Hong et al., 2023; Wu et al., 2024; Chen et al., 2023) provide agent orchestration infrastructure but do not address structured output validation or domain-specific invariant enforcement required for regulatory extraction.

A.3 Self-Correction in Language Models

Self-Refine (Madaan et al., 2023) established iterative self-correction through self-generated feedback, while Reflexion (Shinn et al., 2023) introduced verbal reinforcement learning to learn from prior failures. ReAct (Yao et al., 2022) interleaves reasoning traces with actions, enabling models to ground outputs in external observations. Chain-of-Thought (Wei et al., 2022), Self-Consistency (Wang et al., 2022b), and Tree of Thoughts (Yao et al., 2023) improve reasoning through deliberation and search. Welleck et al. (2022) formalized self-correction as a learnable generation strategy.

A critical finding by Huang et al. (2023) demonstrated that LLMs struggle to self-correct reasoning without external feedback signals. Pan et al. (2024) survey the landscape, categorizing correction strategies by when they occur (training-time, generation-time, and post-hoc) and whether feedback is self-generated or externally sourced. CRITIC (Gou et al., 2023) operationalized this insight by introducing a verify→correct→verify cycle that grounds correction in external tool interaction (search engines, code interpreters), achieving consistent improvements across QA, math, and toxicity reduction at ICLR 2024.

In the legal domain, L4M (Chen et al., 2025) is the closest system to ours in combining multi-agent architecture with iterative correction. L4M embeds prosecutor- and defense-aligned LLM agents inside a logic-based control loop for case adjudication: agents extract facts and statutes into typed schemas, an SMT solver verifies soundness, and unsatisfiable cores trigger iterative self-critique until a consistent solution is reached. L4M surpasses GPT-4o on Chinese criminal law judgment prediction through this combination of symbolic verification and neural reasoning. However, L4M targets adjudication (determining case outcomes) rather than information extraction, and its correction mechanism relies on formal logic solvers that are not applicable to the structural and semantic errors that arise in regulatory extraction (missing criteria, incorrect hierarchical nesting, unresolved references).

Notably, among the extraction-focused systems reviewed above, only AgenticIE and P2T incorporate correction loops. AgenticIE re-prompts failed extractions with failure feedback, and P2T applies an LLM judge to score and repair individual rules. However, neither grounds its correction in the original source document: AgenticIE's repair is triggered by tool-execution failures rather than source comparison, and P2T's judge evaluates rule quality without cross-checking against the regulation text. Our ODR mechanism extends the CRITIC principle to structured regulatory extraction. Like CRITIC, ODR grounds correction in external evidence (the source HTML document); like L4M, it uses iterative refinement driven by external verification. Unlike both, ODR operates within a multi-stage extraction pipeline where each agent has a domain-specific issue taxonomy (structural, semantic, completeness, consistency), a structured diagnosis step that identifies root causes and recommends actions, and a checkpoint mechanism that preserves the best intermediate result across iterations.

A.4 Retrieval-Augmented Generation

RAG (Lewis et al., 2020) addresses LLM knowledge limitations by grounding generation in retrieved documents. Dense passage retrieval (Karpukhin et al., 2020) and sentence embeddings (Reimers and Gurevych, 2019) enable semantic search, while Gao et al. (2023) catalogue advances across retrieval, augmentation, and generation strategies. Recent work has moved toward *adaptive* retrieval: FLARE (Jiang et al., 2023) triggers retrieval when the generator's token-level confidence drops below a threshold, Self-RAG (Asai et al., 2023) trains the model itself to decide when retrieval is needed via learned self-reflection tokens, and CRAG (Yan et al., 2024) introduces a corrective retrieval mechanism that evaluates retrieval quality and triggers refined searches when initial results are insufficient. In the legal domain, RAG has been applied to question answering over case law and legislation, though primarily for single-query retrieval rather than structured extraction.

Our criterion-conditioned RAG differs from standard approaches in three key respects. First, each criterion serves as an independent retrieval query, fusing late-interaction token-level matching (Khattab and Zaharia, 2020) with sparse lexical signals via reciprocal rank fusion. Second, retrieval is followed by *criterion-focused summarization* that extracts only the thresholds, definitions, and conditions relevant to that specific compliance requirement, rather than returning generic document summaries. Third, and most distinctively, the resolved content is *persisted inline* within each criterion's structured output, producing self-contained representations. While standard RAG retrieves context to assist generation and then discards it, our pipeline embeds the resolved content as a permanent component of the output, so downstream consumers require no access to the original reference documents.

A.5 Markup-Aware Models and Graph-Based Retrieval

Markup-aware language models such as MarkupLM (Li et al., 2022) and WebFormer (Wang et al., 2022a) jointly encode DOM structure and text to recover semantic roles in visually rich documents. These methods target settings where the markup carries the missing semantic signal. EUR-Lex HTML is structurally clean, but its markup does not encode the underlying logical hierarchy: as shown in Section 3.3 (Stage 1) and Appendix B, two mutually exclusive compliance pathways routinely share a single `<ol>` with no markup cue between them. The parsing challenge is therefore semantic rather than markup-driven, which motivates the typographic and semantic anchoring used in Stage 1 instead of a markup-aware backbone.

Graph-based retrieval, exemplified by GraphRAG (Edge et al., 2024), retrieves over typed knowledge-graph edges rather than embedding similarity alone. Stages 5 and 6 of REGREACT follow this principle by constraining dependency resolution and footnote attribution to the six typed edges of the criterion graph (Section 3.5). The graph is built per-activity from the regulatory text, so the typed structure constrains retrieval during construction rather than against a pre-existing knowledge base.

B Example: End-to-End Extraction

This appendix presents a complete input–output pair for Activity CCM 4.29 (*Electricity generation from fossil gaseous fuels*), one of the most structurally complex activities in the EU Taxonomy. The input (Figure 3) shows the regulation text as rendered for human readers; the output (Figures 4–7, spanning four pages) shows the structured JSON produced by REGREACT.

Key phenomena demonstrated:

- **Semantic pathway detection**: criterion 1 contains two mutually exclusive pathways—`1(a–e)` (low-emission) vs. `1(f)` (transitional)—that share a single `<ol>` in the HTML but require an OR container with two AND groups.
- **Cross-reference correction**: the regulation text says "point 1(b)" in multiple places, but semantic analysis reveals these refer to `1(f)`; the output records `[CORR FROM:1(b) TO:1(f)]`.
- **Threshold inheritance**: verification sub-criterion `1(f).Verification(a)` inherits quantitative thresholds from `1(f)(a)` via `[THRESHOLD_FROM:1(f)(a)]`.
- **Unnumbered paragraph anchoring**: two paragraphs between criteria 1 and 2 have no numbering in the source; they are anchored as `1(f).Verification` and `1(f).BackgroundInformation` with semantic type tags.
- **Dependencies**: ancillary nodes carry explicit dependency clauses (e.g., `1(f).Verification` depends on pathway

`1(f)` being chosen).

- **OR-linked thresholds with periods**: criterion `1(f)(a)` has two alternative quantitative limits connected by OR logic, one with a 20-year bounded period.
- **Footnote structuring**: footnote 230 on criterion `1(f)(g)` is decomposed into 13 structured items distinguishing `must_be_fetched` from `citation_only` references.
- **External references**: multiple criteria link to EU legislation with normalized CELEX identifiers and `must_be_fetched` / `citation_only` typing.

EU Taxonomy — Activity CCM 4.29: Electricity generation from fossil gaseous fuels
Substantial contribution criteria (source: Delegated Regulation (EU) 2022/1214, Annex I)

1. The activity meets either of the following criteria:

(a) the life-cycle GHG emissions from the generation of electricity using fossil gaseous fuels are lower than 100 g CO_2e/kWh.

(b) Life-cycle GHG emissions are calculated based on project-specific data, where available, using Recommendation 2013/179/EU or, alternatively, using ISO 14067:2018 or ISO 14064-1:2018.

(c) Quantified life-cycle GHG emissions are verified by an independent third party.

(d) Where facilities incorporate any form of abatement, including carbon capture or use of renewable or low-carbon gases, that abatement activity complies with the criteria set out in the relevant Section of this Annex, where applicable.

(e) Where the CO_2 that would otherwise be emitted from the electricity generation process is captured for the purpose of underground storage, the CO_2 is transported and stored underground, in accordance with the technical screening criteria set out in Sections 5.11 and 5.12 of this Annex.

(f) facilities for which the construction permit is granted by 31 December 2030 comply with all of the following:

(a) direct GHG emissions of the activity are lower than 270 g CO_2e/kWh of the output energy, or annual direct GHG emissions of the activity do not exceed an average of 550 kg CO_2e/kW of the facility's capacity over 20 years;

(b) the power to be replaced cannot be generated from renewable energy sources, based on a comparative assessment with the most cost-effective and technically feasible renewable alternative for the same capacity identified; the result of this comparative assessment is published and is subject to a stakeholder consultation;

(c) the activity replaces an existing high emitting electricity generation activity that uses solid or liquid fossil fuels;

(d) the newly installed production capacity does not exceed the capacity of the replaced facility by more than 15 %;

(e) the facility is designed and constructed to use renewable and/or low-carbon gaseous fuels and the switch to full use of renewable and/or low-carbon gaseous fuels takes place by 31 December 2035, with a commitment and verifiable plan approved by the management body of the undertaking;

(f) the replacement leads to a reduction in emissions of at least 55 % GHG over the lifetime of the newly installed production capacity;

(g) where the activity takes place on the territory of a Member State in which coal is used for energy generation, that Member State has committed to phase-out the use of energy generation from coal and has reported this in its integrated national energy and climate plan referred to in Article 3 of Regulation (EU) 2018/1999 of the European Parliament and of the Council[(230)] or in another instrument.

Compliance with the criteria referred to in point 1(b) is verified by an independent third party. The independent third-party verifier has the necessary resources and expertise to perform such verification. The independent third party verifier does not have any conflict of interest with the owner or the funder, and is not involved in the development or operation of the activity. The independent third party verifier carries out diligently the verification of compliance with the technical screening criteria. In particular, every year the independent third party publishes and transmits to the Commission a report:

(a) certifying the level of direct GHG emissions referred to in point 1(b)(i);

(b) where applicable, assessing whether annual direct GHG emissions of the activity are on a credible trajectory to comply with the average threshold over 20 years referred to in point 1(b)(i);

(c) assessing whether the activity is on a credible trajectory to comply with point 1(b)(v).

When undertaking the assessment referred to in point 1(b), the independent third party verifier takes into account in particular the planned annual direct GHG emissions for each year of the trajectory, realised annual direct GHG emissions, planned and realised operating hours, and planned and realised use of renewable or low carbon gases.

On the basis of the reports transmitted to it, the Commission may address an opinion to the relevant operators. The Commission shall take those reports into account when performing the review referred to in Article 19(5) of Regulation (EU) 2020/852.

2. The activity meets either of the following criteria:

(a) at construction, measurement equipment for monitoring of physical emissions, such as those from methane leakage, is installed or a leak detection and repair programme is introduced;

(b) at operation, physical measurement of emissions are reported and leak is eliminated.

3. Where the activity blends fossil gaseous fuels with gaseous or liquid biofuels, the agricultural biomass used for the production of the biofuels complies with the criteria laid down in Article 29, paragraphs 2 to 5, of Directive (EU) 2018/2001 while forest biomass complies with the criteria laid down in Article 29, paragraphs 6 and 7, of that Directive.

[(230)] Regulation (EU) 2018/1999 of the European Parliament and of the Council of 11 December 2018 on the governance of the energy union and climate action, amending regulations (EC) No 663/2009 and (EC) No 715/2009 [...], Council Directives 2009/119/EC and (EU) 2015/652 and repealing Regulation (EU) No 525/2013 (OJ L 328, 21.12.2018, p. 1).

Figure 3: Input regulation text for Activity CCM 4.29, rendered as it appears to human readers. Note: the two `<ol>` elements (a–f and a–g) share no explicit grouping in the HTML source; the semantic pathway split (`1(a–e)` vs. `1(f)`) must be inferred. The unnumbered paragraphs between criteria 1 and 2 (starting "Compliance with the criteria…") have no identifiers in the source.

Annotation Legend

Pathway Detection Inferred structural grouping absent from source HTML. **Footnote Structuring** Items classified as `must_be_fetched` vs. `citation_only`.

Threshold Inheritance Values propagated from a parent criterion. **Cross-Ref Correction** Erroneous cross-references detected and corrected by the ODR stage.

Anchored Paragraph Unnumbered source paragraph assigned a synthetic criterion ID by the pipeline.

RAG Enrichment Referenced content retrieved, summarized, and embedded inline by the RAG sub-pipeline.

Structured JSON Output — Page 1 of 4: Root and Low-Emission Pathway 1(a–e)


```
{
  "criterion_id": "ROOT",
  "category": "Qualitative",
  "applicability": "Mandatory",
  "tags": [],
  "rule_summary": "The activity must satisfy all top-level criteria (1, 2, and 3).",
  "evaluation_logic": "AND",
  "n_required": null,
  "threshold": null,
  "references": null,
  "dependencies": null,
  "footnotes": [],
  "sub_criteria": [
    {
      "criterion_id": "1",
      "category": "Qualitative",
      "applicability": "Mandatory",
      "tags": [],
      "rule_summary": "...either the low-emission pathway (a-e) or the transitional pathway (f).",
      "evaluation_logic": "OR", // Two mutually exclusive pathways
      "n_required": null,
      "threshold": null,
      "references": null,
      "dependencies": null,
      "footnotes": [],
      "sub_criteria": [
        {
          "criterion_id": "1(a-e)", // Inferred AND group
          "category": "Qualitative",
          "applicability": "Mandatory",
          "tags": [],
          "rule_summary": "Groups 1(a)-1(e) as cumulative low-emission conditions forming one alternative pathway.",
          "evaluation_logic": "AND",
          "n_required": null,
          "threshold": null,
          "references": null,
          "dependencies": null,
          "footnotes": [],
          "sub_criteria": [
            {
              "criterion_id": "1(a)",
              "category": "Quantitative", // Quantitative: has threshold
              "applicability": "Mandatory",
              "tags": [],
              "rule_summary": "Life-cycle GHG emissions must be below 100 g CO2e/kWh.",
              "evaluation_logic": "LEAF",
              "n_required": null,
              "threshold": {
                "quantitative": {
                  "logic": null,
                  "items": [
                    { "metric": "Life-cycle GHG emissions",
                      "operator": "<", "value": 100,
                      "unit": "g CO2e/kWh", "period": null }
                  ]
                },
                "temporal": null
              },
              "references": null, "dependencies": null,
              "footnotes": [], "sub_criteria": []
            },
            {
              "criterion_id": "1(b)",
              "category": "Qualitative",
```

```
      "applicability": "Mandatory",
      "tags": ["Methodology"],
      "rule_summary": "Emissions calculated using Rec. 2013/179/EU or ISO 14067:2018 or ISO 14064-1:2018.",
      "evaluation_logic": "LEAF",
      "n_required": null,
      "threshold": null,
      "references": {
        "logic": "OR",            // OR-linked external references
        "sources": [
          { "text": "Recommendation 2013/179/EU",
            "type": "must_be_fetched", "celex_id": "32013H0179",
            "enrichment": {
              "status": "retrieved",
              "summary": "Provides the PEF method for
                  measuring life-cycle environmental
                  performance including GHG emissions.",
              "key_facts": ["PEF life-cycle method",
                  "Environmental performance of products"],
              "thresholds": [],
              "confidence": 0.85
            }
          },
          { "text": "ISO 14067:2018",
            "type": "must_be_fetched", "celex_id": null,
            "enrichment": {"status": "paywalled"}
          },
          { "text": "ISO 14064-1:2018",
            "type": "must_be_fetched", "celex_id": null,
            "enrichment": {"status": "paywalled"}
          }
        ]
      },
      "dependencies": null, "footnotes": [], "sub_criteria": []
    },
    {
      "criterion_id": "1(c)",
      "category": "Qualitative", "applicability": "Mandatory",
      "tags": ["Verification"],
      "rule_summary": "Life-cycle GHG emissions verified by independent third party.",
      "evaluation_logic": "LEAF",
      "n_required": null, "threshold": null, "references": null,
      "dependencies": null, "footnotes": [], "sub_criteria": []
    },
    {
      "criterion_id": "1(d)",
      "category": "Qualitative", "applicability": "Conditional",
      "tags": ["Methodology"],
      "rule_summary": "If abatement is used, it must comply with the relevant Section of this Annex.",
      "evaluation_logic": "LEAF",
      "n_required": null, "threshold": null,
      "references": { "logic": null, "sources": [
        { "text": "the relevant Section of this Annex",
          "type": "must_be_fetched", "celex_id": null,
          "enrichment": {"status": "retrieved",
            "summary": "Covers gas networks for
                hydrogen and low-carbon gases,
                including new and converted
                networks.",
            "key_facts": ["Networks for hydrogen
                or low-carbon gases",
                "Leak detection for methane"],
            "thresholds": [],
            "confidence": 0.85}
        }
      ]},
      "dependencies": null, "footnotes": [], "sub_criteria": []
    },
    {
      "criterion_id": "1(e)",
      "category": "Qualitative", "applicability": "Conditional",
      "tags": ["Methodology"],
      "rule_summary": "If CO2 is captured, transport and storage must comply with Sections 5.11 and 5.12.",
      "evaluation_logic": "LEAF",
      "n_required": null, "threshold": null,
```

```
          "references": { "logic": "AND", "sources": [
            { "text": "Section 5.11 of this Annex",
              "type": "must_be_fetched", "celex_id": null,
              "enrichment": {"status": "retrieved",
                "summary": "CO2 transport must limit
                  leakage to 0.5% of mass transported.",
                "key_facts": ["CO2 leakage limit",
                  "Monitoring verified by third party"],
                "thresholds": ["0.5%"],
                "confidence": 0.90}
            },
            { "text": "Section 5.12 of this Annex",
              "type": "must_be_fetched", "celex_id": null,
              "enrichment": {"status": "retrieved",
                "summary": "Storage sites require
                  characterisation, leakage detection,
                  and compliance with Directive
                  2009/31/EC.",
                "key_facts": ["Storage site
                  assessment", "Leakage detection
                  and monitoring"],
                "thresholds": [],
                "confidence": 0.90}
            }
          ]},
          "dependencies": null, "footnotes": [], "sub_criteria": []
        }
      ]
    },
    // continued on next page: transitional pathway 1(f)
```


Figure 4: Structured JSON output (Page 1 of 4). The root AND container joins three top-level criteria. Criterion 1 uses OR to select between the two pathways. The inferred AND group `1(a-e)` does not exist in the source HTML—it is created by semantic pathway detection. Criterion `1(b)` demonstrates inline RAG enrichment: the EU legal act is *retrieved* with a criterion-conditioned summary, while ISO standards are marked *paywalled*. Criterion `1(e)` shows a *retrieved* internal cross-reference with an extracted threshold (`0.5%`). Fields `verbatim_text`, `link`, and `link_status` are omitted throughout for brevity.

Structured JSON Output — Page 2 of 4: Transitional Pathway 1(f)


```
    // continued: second alternative under criterion "1"
    {
      "criterion_id": "1(f)",         // Transitional pathway
      "category": "Qualitative",
      "applicability": "Mandatory",
      "tags": [],
      "rule_summary": "Facilities permitted by 31 Dec 2030 must comply with all seven sub-criteria.",
      "evaluation_logic": "AND",
      "n_required": null,
      "threshold": {
        "quantitative": null,
        "temporal": {                 // Deadline on parent container
          "logic": null,
          "items": [{ "type": "deadline", "date": "2030-12-31" }]
        }
      },
      "references": null, "dependencies": null, "footnotes": [],
      "sub_criteria": [
        {
          "criterion_id": "1(f)(a)",
          "category": "Quantitative", "applicability": "Mandatory",
          "tags": [],
          "rule_summary": "Direct GHG < 270 g CO2e/kWh, or annual average <= 550 kg CO2e/kW over 20 years.",
          "evaluation_logic": "LEAF",
          "n_required": null,
          "threshold": {
            "quantitative": {
              "logic": "OR",          // Two alternative limits
              "items": [
```

```
          { "metric": "Direct GHG emissions",
            "operator": "<", "value": 270,
            "unit": "g CO2e/kWh", "period": null },
          { "metric": "Average annual direct GHG emissions",
            "operator": "<=", "value": 550,
            "unit": "kg CO2e/kW",
            "period": { "type": "bounded", "value": 20, "unit": "years" } }
        ]
      },
      "temporal": null
    },
    "references": null, "dependencies": null,
    "footnotes": [], "sub_criteria": []
  },
  {
    "criterion_id": "1(f)(b)",
    "category": "Qualitative", "applicability": "Mandatory",
    "tags": ["Assessment"],
    "rule_summary": "Published comparative assessment must show renewables cannot replace the power.",
    "evaluation_logic": "LEAF",
    "n_required": null, "threshold": null, "references": null,
    "dependencies": null, "footnotes": [], "sub_criteria": []
  },
  {
    "criterion_id": "1(f)(c)",
    "category": "Qualitative", "applicability": "Mandatory",
    "tags": ["Replacement"],
    "rule_summary": "Must replace an existing high-emission facility using solid or liquid fossil fuels.",
    "evaluation_logic": "LEAF",
    "n_required": null, "threshold": null, "references": null,
    "dependencies": null, "footnotes": [], "sub_criteria": []
  },
  {
    "criterion_id": "1(f)(d)",
    "category": "Quantitative", "applicability": "Mandatory",
    "tags": ["Replacement"],
    "rule_summary": "New capacity must not exceed replaced capacity by more than 15%.",
    "evaluation_logic": "LEAF",
    "n_required": null,
    "threshold": {
      "quantitative": { "logic": null, "items": [
        { "metric": "Capacity increase",
          "operator": "<=", "value": 15, "unit": "%", "period": null }
      ]},
      "temporal": null
    },
    "references": null, "dependencies": null,
    "footnotes": [], "sub_criteria": []
  },
  {
    "criterion_id": "1(f)(e)",
    "category": "Qualitative", "applicability": "Mandatory",
    "tags": ["Commitment"],
    "rule_summary": "Full switch to renewable/low-carbon fuels by 31 Dec 2035 with approved plan.",
    "evaluation_logic": "LEAF",
    "n_required": null,
    "threshold": {
      "quantitative": null,
      "temporal": { "logic": null,
        "items": [{ "type": "deadline", "date": "2035-12-31" }]
      }
    },
    "references": null, "dependencies": null,
    "footnotes": [], "sub_criteria": []
  },
  {
    "criterion_id": "1(f)(f)",
    "category": "Quantitative", "applicability": "Mandatory",
    "tags": [],
    "rule_summary": "Replacement must reduce lifetime GHG by >= 55%.",
    "evaluation_logic": "LEAF",
    "n_required": null,
    "threshold": {
      "quantitative": { "logic": null, "items": [
        { "metric": "GHG emission reduction",
          "operator": ">=", "value": 55, "unit": "%",
          "period": { "type": "lifetime" } }
      ]},
      "temporal": null
    },
    "references": null, "dependencies": null,
    "footnotes": [], "sub_criteria": []
  },
  {
    "criterion_id": "1(f)(g)",
    "category": "Qualitative", "applicability": "Conditional",
```

```
          "tags": ["Commitment"],
          "rule_summary": "If in coal-using Member State, that state must have committed to phase out coal.",
          "evaluation_logic": "LEAF",
          "n_required": null, "threshold": null,
          "references": { "logic": "OR", "sources": [
            { "text": "Article 3 of Regulation (EU) 2018/1999",
              "type": "must_be_fetched", "celex_id": "32018R1999",
              "enrichment": {"status": "retrieved",
                "summary": "Requires each Member State
                  to notify the Commission of an
                  integrated national energy and
                  climate plan.",
                "key_facts": ["Integrated national
                  energy and climate plans",
                  "Plans publicly available"],
                "thresholds": [],
                "confidence": 0.85}
            },
            { "text": "another instrument",
              "type": "must_be_fetched", "celex_id": null,
              "enrichment": {"status": "skipped"}
            }
          ]},
          "dependencies": null,
          "footnotes": [{
            "id": "fn-230",
            "categories": ["Legal Reference"],
            "items": [
              { "kind": "EU Legal Act",
                "title": "Regulation (EU) 2018/1999",
                "celex_id": "32018R1999",
                "type": "must_be_fetched", "oj": "OJ L 328, 21.12.2018, p. 1",
                "enrichment": {"status": "retrieved",
                  "summary": "Sets national renewable
                    energy targets within integrated
                    energy and climate plans.",
                  "key_facts": ["National renewable
                    energy targets", "Commission
                    assesses plan ambition"],
                  "thresholds": [],
                  "confidence": 0.60}
              },
              { "kind": "EU Legal Act",
                "title": "Regulation (EC) No 663/2009",
                "type": "citation_only" },
              { "kind": "EU Legal Act",
                "title": "Directive 94/22/EC",
                "type": "citation_only" }
              // ... +10 more "citation_only" items omitted
            ],
            "definitions": [], "notes": []
          }],
          "sub_criteria": []
        }
      ]
    },
    // continued on next page: anchored paragraphs
```


Figure 5: Structured JSON output (Page 2 of 4). Transitional pathway `1(f)` shows: temporal deadline on the parent container; OR-linked quantitative thresholds with bounded periods on `1(f)(a)`; a lifetime period on `1(f)(f)`; structured footnote 230 with 13 items distinguishing `must_be_fetched` from `citation_only`, where the `must_be_fetched` item also carries enrichment; and `1(f)(g)` enrichment showing a *retrieved* EU legal act alongside a *skipped* vague reference.

Structured JSON Output — Page 3 of 4: Anchored Paragraphs and Corrections


```
// continued: unnumbered paragraphs anchored under criterion "1"
{
  "criterion_id": "1(f).Verification", // Anchored paragraph
  "category": "Qualitative", "applicability": "Mandatory",
  "tags": ["Verification"],
  "rule_summary": "An independent third party must annually verify compliance with the
    transitional pathway criteria (1(f)) and submit a report to the Commission.
    [CORR FROM:1(b) TO:1(f) REASON:crossref] ",
  "evaluation_logic": "AND",
  "n_required": null, "threshold": null, "references": null,
  "dependencies": {
    "condition_summary": "This verification requirement applies only to activities complying via the transitional pathway.",
    "min_conditions_to_meet": 1,
    "clauses": [
      { "criterion_id": "1(f)", "status": "Affirmation" }
    ]
  },
  "footnotes": [],
  "sub_criteria": [
    {
      "criterion_id": "1(f).Verification(a)",
      "category": "Quantitative", "applicability": "Mandatory",
      "tags": ["Verification"],
      "rule_summary": "Certify direct GHG < 270 g CO2e/kWh or on track for 550 kg CO2e/kW
        over 20 years. [THRESHOLD_FROM:1(f)(a)] [CORR FROM:1(b)(i) TO:1(f)(a)] ",
      "evaluation_logic": "LEAF",
      "n_required": null,
      "threshold": {
        "quantitative": {
          "logic": "OR",
          "items": [
            { "metric": "Direct GHG emissions",
              "operator": "<", "value": 270,
              "unit": "g CO2e/kWh", "period": null },
            { "metric": "Average annual direct GHG emissions",
              "operator": "<=", "value": 550,
              "unit": "kg CO2e/kW",
              "period": { "type": "bounded", "value": 20, "unit": "years" } }
          ]
        },
        "temporal": null
      },
      "references": null, "dependencies": null,
      "footnotes": [], "sub_criteria": []
    },
    {
      "criterion_id": "1(f).Verification(b)",
      "category": "Quantitative", "applicability": "Conditional",
      "tags": ["Verification"],
      "rule_summary": "Assess trajectory to meet 550 kg CO2e/kW threshold.
        [THRESHOLD_FROM:1(f)(a)] [CORR FROM:1(b)(i) TO:1(f)(a)] ",
      "evaluation_logic": "LEAF",
      "n_required": null,
      "threshold": {
        "quantitative": { "logic": null, "items": [
          { "metric": "Average annual direct GHG emissions",
            "operator": "<=", "value": 550,
            "unit": "kg CO2e/kW",
            "period": { "type": "bounded", "value": 20, "unit": "years" } }
        ]},
        "temporal": null
      },
      "references": null, "dependencies": null,
      "footnotes": [], "sub_criteria": []
    },
    {
      "criterion_id": "1(f).Verification(c)",
      "category": "Qualitative", "applicability": "Mandatory",
      "tags": ["Verification"],
      "rule_summary": "Assess credible trajectory to complete fuel switch by 2035.
        [THRESHOLD_FROM:1(f)(e)] [CORR FROM:1(b)(v) TO:1(f)(e)] ",
      "evaluation_logic": "LEAF",
      "n_required": null,
      "threshold": {
        "quantitative": null,
        "temporal": { "logic": null,
          "items": [{ "type": "deadline", "date": "2035-12-31" }]
        }
```

```
        },
        "references": null, "dependencies": null,
        "footnotes": [], "sub_criteria": []
      }
    ]
  },
  // continued on next page: BackgroundInformation, criteria 2 and 3
```

Figure 6: Structured JSON output (Page 3 of 4). The `1(f).Verification` anchored paragraph and its sub-criteria are unnumbered in the source—the pipeline assigns them synthetic IDs. `[CORR]` corrections fix erroneous "point 1(b)" references. `[THRESHOLD_FROM]` propagates values from `1(f)(a)` and deadlines from `1(f)(e)`.

Structured JSON Output — Page 4 of 4: Background, Criteria 2–3, and Closing

```
  // continued: second anchored paragraph group under criterion "1"
  {
    "criterion_id": "1(f).BackgroundInformation",
    "category": "Qualitative", "applicability": "Mandatory",
    "tags": ["BackgroundInformation"],
    "rule_summary": "Groups background context paragraphs for the transitional pathway (1(f)) verification and review.",
    "evaluation_logic": "AND",
    "n_required": null, "threshold": null, "references": null,
    "dependencies": {
      "condition_summary": "Applies to the transitional pathway.",
      "min_conditions_to_meet": 1,
      "clauses": [
        { "criterion_id": "1(f)", "status": "Affirmation" }
      ]
    },
    "footnotes": [],
    "sub_criteria": [
      {
        "criterion_id": "1(f).BackgroundInformation(a)",
        "category": "Qualitative", "applicability": "Mandatory",
        "tags": ["BackgroundInformation"],
        "rule_summary": "Verifier considers planned vs realised emissions, operating hours,
          and use of renewable gases. [CORR FROM:1(b) TO:1(f)]",
        "evaluation_logic": "LEAF",
        "n_required": null, "threshold": null, "references": null,
        "dependencies": null, "footnotes": [], "sub_criteria": []
      },
      {
        "criterion_id": "1(f).BackgroundInformation(b)",
        "category": "Qualitative", "applicability": "Conditional",
        "tags": ["BackgroundInformation"],
        "rule_summary": "Commission may address opinions to operators and considers reports in its Article 19(5) review.",
        "evaluation_logic": "LEAF",
        "n_required": null, "threshold": null,
        "references": { "logic": null, "sources": [
          { "text": "Article 19(5) of Regulation (EU) 2020/852",
            "type": "must_be_fetched", "celex_id": "32020R0852",
            "enrichment": {"status": "retrieved",
              "summary": "Commission must review
                technical screening criteria at
                least every three years.",
              "key_facts": ["Review every three
                years", "Assess impact on
                capital markets"],
              "thresholds": [],
              "confidence": 0.85}
          }
        ]},
        "dependencies": {
          "condition_summary": "Applies when verification reports are submitted to the Commission.",
          "min_conditions_to_meet": 1,
          "clauses": [
            { "criterion_id": "1(f).Verification", "status": "Affirmation" }
          ]
        },
        "footnotes": [], "sub_criteria": []
      }
    ]
```

```
      }
    ]
  },
  {
    "criterion_id": "2",
    "category": "Qualitative", "applicability": "Mandatory",
    "tags": [],
    "rule_summary": "The activity must implement emission monitoring and repair measures during construction or operation.",
    "evaluation_logic": "OR",
    "n_required": null, "threshold": null, "references": null,
    "dependencies": null, "footnotes": [],
    "sub_criteria": [
      {
        "criterion_id": "2(a)",
        "category": "Qualitative", "applicability": "Mandatory",
        "tags": ["Methodology"],
        "rule_summary": "At construction: install emission monitoring equipment or introduce a leak detection program.",
        "evaluation_logic": "LEAF",
        "n_required": null, "threshold": null, "references": null,
        "dependencies": null, "footnotes": [], "sub_criteria": []
      },
      {
        "criterion_id": "2(b)",
        "category": "Qualitative", "applicability": "Mandatory",
        "tags": ["Methodology"],
        "rule_summary": "At operation: report physical emissions and eliminate any detected leaks.",
        "evaluation_logic": "LEAF",
        "n_required": null, "threshold": null, "references": null,
        "dependencies": null, "footnotes": [], "sub_criteria": []
      }
    ]
  },
  {
    "criterion_id": "3",
    "category": "Qualitative", "applicability": "Conditional",
    "tags": ["Methodology"],
    "rule_summary": "If blending with biofuels: agricultural biomass must meet Art. 29(2-5) and forest biomass Art. 29(6-7) of
        ↪ Directive (EU) 2018/2001.",
    "evaluation_logic": "LEAF",
    "n_required": null, "threshold": null,
    "references": {
      "logic": "AND",
      "sources": [
        { "text": "Article 29, paragraphs 2 to 5, of Directive (EU) 2018/2001",
          "type": "must_be_fetched", "celex_id": "32018L2001",
          "enrichment": {"status": "retrieved",
            "summary": "Defines renewable energy
                sources including biomass and
                biogas.",
            "key_facts": ["Renewable energy
                sources defined",
                "Biomass and biogas included"],
            "thresholds": [],
            "confidence": 0.60}
        },
        { "text": "Article 29, paragraphs 6 and 7, of that Directive",
          "type": "must_be_fetched", "celex_id": "32018L2001",
          "enrichment": {"status": "retrieved",
            "summary": "Defines renewable energy
                sources including biomass and
                biogas.",
            "key_facts": ["Renewable energy
                sources defined",
                "Biomass and biogas included"],
            "thresholds": [],
            "confidence": 0.60}
        }
      ]
    },
    "dependencies": null, "footnotes": [], "sub_criteria": []
  }
]
}
```

Figure 7: Structured JSON output (Page 4 of 4). The `BackgroundInformation` anchored group likewise has no numbering in the source. Enrichment on `BackgroundInfo(b)` and Criterion 3 shows *retrieved* summaries with varying confidence levels.

C ODR Self-Correction Details

The ODR observation step categorizes detected issues into four types:

- *Structural*: format violations or missing required fields;
- *Semantic*: operator inconsistent with text (e.g., "$>$" when text says "at most");
- *Completeness*: criteria or thresholds present in the source but absent from the output;
- *Consistency*: cross-field contradictions (e.g., category marked Quantitative but no threshold data).

Each issue carries a severity level, and confidence is computed via a penalty model that deducts from a base score proportionally to issue count and severity, with the final score clamped to $[0, 1]$.

D Prompt Templates

All prompts instruct the LLM to return JSON-only output. Formatting boilerplate ("Return ONLY valid JSON", output schema definitions) is omitted below for brevity; complete prompts are available in our repository. Prompts are grouped by pipeline stage (§3.3).

D.1 Stage 1 (Structural Parser): Evaluation Logic Inference

Stage 1 (Structural Parser): Evaluation Logic Inference

Task: Determine the logical relationship between items following the chapeau text.
Input: `{chapeau_text}`

3-Step Semantic Reasoning (apply all three steps):
Step 1 — Simultaneity: Can ALL children be satisfied at the same time?
- If children describe conditions that CAN coexist → AND is possible
- If children describe MUTUALLY EXCLUSIVE alternatives → must be OR

Step 2 — Sufficiency: Is satisfying ONE child sufficient for compliance?
- ONE child = compliant → OR
- Need more → continue to Step 3

Step 3 — Completeness: Are ALL children required?
- ALL required → AND (cumulative)
- Specific count → N_OF_K
- Default (unclear) → AND (regulatory conservative default)

Special Pattern — Conditional Tiering: When the chapeau introduces values followed by a colon, and children represent conditional categories (different values for different situations based on mutually exclusive attributes such as size, count, or type), use OR — an entity can only be in ONE category at a time.

Examples:
`A:` "The activity complies with one of the following criteria"
Step 1: check → Step 2: "one of" = ONE is sufficient → **OR**
`B:` "The activity meets all of the following requirements"
Step 1: can coexist → Step 2: "all of" = ONE is not sufficient → Step 3: "all" = ALL required → **AND**
`C:` "Facilities for which approval is granted by 31 December 2027"
Step 1: scope statement, not alternatives → Step 3: requirements apply to these facilities → **AND**
`D:` "The activity demonstrates compliance through either of the following pathways"
Step 2: "either of" = ONE pathway is sufficient → **OR**
`E:` "At least equivalent to the applicable Union legislation"
Comparison benchmark, not selection between children → **AND**
`F:` "The percentage contribution is at least equivalent to:"
Children have conditional phrases ("where agreement includes 1 establishment", "where includes 2–10") — mutually exclusive tiers → **OR**

D.2 Stage 2 (Threshold Extractor): Quantitative Threshold Extraction

Stage 2 (Threshold Extractor): Quantitative Threshold Extraction

Task: Extract numeric performance thresholds (NOT dates/deadlines) from regulatory text.
Input: `{text}`

Metric Naming: Use the COMPLETE qualifier chain from source text:
- "life-cycle GHG emissions" → `"Life-cycle GHG emissions"` (NOT just "GHG emissions")
- "direct GHG emissions" → `"Direct GHG emissions"`
- "average annual direct GHG emissions" → include "Average annual"
- For CHANGES: "does not exceed X by more than 15%" → `"Capacity increase"`
- For REDUCTIONS: "reduction of at least 55%" → `"GHG emission reduction"`
- Causal construction: "X leads to a reduction of Y" → metric is the MEASURED QUANTITY (Y), not the subject (X)
- Metric must be a clean noun phrase; always preserve modifiers (life-cycle, direct, annual, average, net, gross)

Operator Detection (Chain-of-Thought):
Step 1 — Compliance direction: What values pass? Lower → < or <=. Higher → > or >=.
Step 2 — Boundary inclusion: "lower than", "below" → < (exclusive). "not exceeding", "at most" → <= (inclusive). "at least", "minimum" → >= (inclusive). "exceeds", "more than" → > (exclusive).
Defaults: Emissions/costs → <=. Contributions/reductions → >=.

Period (measurement timeframe, NOT deadline): `null`: no period. `lifetime`: "over the lifetime". `bounded`: "over 20 years" → `{type: bounded,`

`value: 20, unit: years}`. Note: "life-cycle" = methodology, NOT lifetime period → use `null`.

Threshold vs Condition — Semantic Test:
Step 1: Does the number constrain WHO/WHAT the rule applies to? → YES = CONDITION (skip).
Step 2: Does achieving this number = passing the requirement? → YES = THRESHOLD (extract).
Examples: "contribution of 1.5% of annual revenue" → compliance test → THRESHOLD. "For establishments with over 50 employees" → constrains WHO → CONDITION (skip).

Compound Alternatives: "X or Y" → `logic="OR"`, extract BOTH.
Example: "emissions below 270g CO2e/kWh, or annual emissions not exceeding 550 kg CO2e/kW over 20 years" → `logic="OR"`, items=[{metric: "Direct GHG emissions", op: $<$, value: 270, unit: "g CO2e/kWh"}, {metric: "Average annual direct GHG emissions", op: $\leq$, value: 550, unit: "kg CO2e/kW", period: {bounded, 20, years}}]

D.3 Stage 2 (Threshold Extractor): Temporal Threshold Extraction

Stage 2 (Threshold Extractor): Temporal Threshold Extraction

Task: Extract dates, deadlines, and recurring intervals (NOT numeric thresholds) from regulatory text.
Input: `{text}`

Types: `deadline`: "by 31 December 2030" → `{type: deadline, date: 2030-12-31}`. `effective_from`: "from 1 January 2026". `effective_until`: "until 31 December 2025". `window`: "between 2026 and 2030". `recurring`: "every five years" → `{type: recurring, interval_value: 5, interval_unit: years}`.

Critical Distinctions:

Pattern	Meaning	Classification
"by 31 December 2030"	Deadline	temporal ✓
"every 5 years"	Recurring interval	temporal ✓
"over 20 years"	Measurement period	quantitative.period ×
"at construction"	Phase name	NOT temporal ×
"minimum of five years"	Duration threshold	quantitative ×

Chain-of-Thought: Step 1: Scan for years, full dates, interval words. Step 2: Classify each — calendar date/deadline → extract; measurement period → skip; phase name → skip. Step 3: Extract from qualifying clauses too: "facilities for which permit is granted by 31 December 2030" → deadline.

D.4 Stage 3 (Content Classifier): Applicability Classification

Stage 3 (Content Classifier): Applicability Classification

Task: Determine if this criterion is Mandatory or Conditional.
Input: `{criterion_id}`, `{verbatim_text}`
Default: Mandatory unless proven otherwise.

Chain-of-Thought Reasoning:
Step 1 — Universal Qualifiers ("where available", "where applicable", "where feasible"):
Three questions: (1) Does this phrase apply to ALL instances universally? (2) Does it modify HOW to comply, not WHETHER to comply? (3) If removed, does the core obligation still exist?
If ALL THREE YES → Mandatory. Example: "calculated based on project-specific data, *where available*" — data source preference, obligation still exists → Mandatory.
If any NO → Conditional. Example: "*where applicable*, assessing whether emissions are on trajectory" — gates a specific pathway → Conditional.

Step 2 — "Where [entity] [verb] X" pattern:
- "Where facilities *incorporate* any form of abatement…" → Conditional
- "Where the CO_2 *is captured* for underground storage…" → Conditional
- "Where the activity *takes place* on territory of a Member State using coal…" → Conditional

These describe SITUATIONS that vary between instances → ALWAYS Conditional.

Step 3 — Semantic Role Analysis:
- **Instance filter** → Conditional: characteristic that VARIES between instances. Test: Can two compliant instances differ on this condition? YES → Conditional.
- **Universal scope** → Mandatory: defining characteristic ALL instances share.

Step 4 — Parent context: AND-parent → default Mandatory. OR-parent → evaluate independently.
Step 5 — Performance vs. Applicability: Performance target (WHAT to achieve, "emissions below 100g") → Mandatory. Applicability threshold (WHO, "facilities over 100 MW") → Conditional.

Temporal context $\neq$ condition: "When undertaking X" = process stage → Mandatory. "When/Where X occurs" = situation → evaluate.
Discretionary actions: "may" → Conditional. "shall" → Mandatory.

D.5 Stage 3 (Content Classifier): Tag Assignment and Rule Summary

Stage 3 (Content Classifier): Tag Assignment and Rule Summary

Task: Assign semantic tags (0–3) and compose a self-contained rule summary.
Input: `{criterion_id}`, `{verbatim_text}`, `{threshold_context}`

Task A — Testability Check: Step 1: What must the operator DO or ACHIEVE? Step 2: Can an auditor create a PASS/FAIL checklist? If NOT testable → tag as BackgroundInformation.

Task B — Tags (3-step semantic test for EACH potential tag):
1. AUDIT TEST: Can an auditor create a PASS/FAIL checklist?
2. OBLIGATION DIRECTION: WHO must act — the operator, or someone else?
3. COMPLIANCE IMPACT: If the operator ignores this, does compliance fail?

ALL THREE must say YES for a requirement tag. If ANY says NO → BackgroundInformation.

Tag definitions:

- **Verification**: Operator must OBTAIN third-party validation. Apply: "verified by independent third party", "certified by accredited body". Do not apply: pure thresholds without explicit verification mention; "verifiable plan" → Commitment.
- **Methodology**: Operator must FOLLOW a specific method/standard/equipment. Apply: "calculated using ISO 14067", "measurement equipment is installed".
- **Commitment**: Operator must have formal plan/declaration approved by management.
- **Assessment**: Operator must conduct internal study/comparison. Apply: "comparative assessment", "stakeholder consultation".
- **Replacement**: PRIMARY semantic role is replacing existing equipment/facility. Do not apply when "replacement" is context for a metric ("leads to 55% reduction" → metric, not replacement).
- **BackgroundInformation**: NON-BINDING context. Apply when: (1) no pass/fail test, (2) subject is NOT the operator, (3) no compliance impact. Examples: "verifier takes into account", "Commission may address an opinion".

Distinction table:

Pattern	Actor	Binding?	Tag
"Compliance IS VERIFIED by…"	Operator	Yes	Verification
"The verifier TAKES INTO ACCOUNT…"	Verifier	No	BackgroundInfo
"IS SUBJECT TO consultation"	Operator	Yes	Assessment
"The Commission MAY address…"	Commission	No	BackgroundInfo

Parent chapeau rule: Parent introducing sub-criteria ("meets either of:") → tags: [].

Task C — Rule Summary: Self-contained summary (1–2 sentences) a compliance professional can understand without the original text.
Structure by classification: Conditional → start with condition. Quantitative → embed ALL thresholds with values and units. Methodology → name the standard.
Bad: "The level of direct GHG emissions must be certified." (missing thresholds). Good: "An independent third party must verify that direct GHG emissions are below 270 g CO_2e/kWh or on track for the 20-year average of 550 kg CO_2e/kW."

D.6 Stage 4 (Reference Extractor): Reference Extraction

Stage 4 (Reference Extractor): Reference Extraction

Task: Extract compliance references that operators must fetch/follow.
Input: `{text}`

Rule 1 — Never abbreviate: Keep "paragraphs" as "paragraphs" (not "p" or "paras"). Copy text EXACTLY.
Rule 2 — Internal vs. external:
- "criterion 1(a)", "point (c)" → SKIP (internal cross-reference, goes to dependencies)
- "Article 3", "ISO 14067" → EXTRACT (external law/standard)
- "Section of this Annex", "Annex A" → EXTRACT (external section/appendix)

Critical distinction: "complies with criteria set out in *Section X*" → REFERENCE (document to fetch). "Where *criterion 1(a)* applies" → SKIP (dependency).
Rule 3 — Legal basis = ONE reference: "plan referred to in Article 3 of Regulation (EU) 2018/1999" → ONE source (Article is part of where the plan is defined).
Rule 4 — Split when independent:
- "Sections 5.11 and 5.12" → TWO independent sections → SPLIT
- "Directives 92/43/EEC and 2009/147/EC" → TWO legal acts → SPLIT
- Multi-subject: "agricultural biomass complies with [Ref_1] while forest biomass complies with [Ref_2]" → SPLIT ("while" = coordination conjunction connecting parallel requirements)
- "Article 3 of Regulation X" → ONE (Article is PART OF the Regulation)

Rule 5 — CELEX IDs: Format: `3[YEAR][TYPE][NUMBER_4_DIGITS]`. Type codes: R=Regulation, L=Directive, D=Decision, H=Recommendation. Number is ALWAYS zero-padded to 4 digits: 179 → "0179", 43 → "0043". Examples: "Recommendation 2013/179/EU" → `32013H0179`. "Directive 92/43/EEC" → `31992L0043`.

Chain-of-Thought (7 steps): (1) Find candidate references. (2) Check if internal → skip. (3) Determine fetch requirement (`must_be_fetched` vs. `citation_only`). (4) Multi-subject detection ("X complies with A while Y complies with B"). (5) Independent reference detection (split "Sections 5.11 and 5.12"). (6) Determine logic (OR: "or", "alternatively"; AND: "and", cumulative). (7) Copy exact text.

Example: "agricultural biomass complies with Article 29, paragraphs 2 to 5, of Directive (EU) 2018/2001 while forest biomass complies with Article 29, paragraphs 6 and 7, of that Directive" → `logic: "AND"`, two sources: [Art. 29(2–5), celex: 32018L2001] and [Art. 29(6–7), celex: 32018L2001]. "while" = parallel compliance, "that Directive" = anaphoric reference.

D.7 Stage 5 (Dependency Resolver): Cross-Reference Disambiguation

Stage 5 (Dependency Resolver): Cross-Reference Disambiguation

Task: Disambiguate an ambiguous cross-reference when multiple criteria could match.
Input: `{current_id}`, `{current_text}`, `{original_id}`, three candidates (original, same-family, parent).

Step 1 — Understand context: What is the current criterion about? (verification, background, requirement?)
Step 2 — Analyze each candidate: Candidate A (`original_id`): cross-family reference to a sibling? Candidate B (`same_family_id`): same-family sub-criterion? Candidate C (`parent_id`): about compliance with the parent overall?
Step 3 — Determine intent:
- If current criterion is a verification/background node, the reference likely means "verify compliance with the parent criterion's requirements" → choose parent
- If current criterion discusses a specific sub-

requirement → choose same-family
- If genuinely cross-referencing another top-level criterion → keep original

Output: `{from: original_id, to: chosen_id, reason: "crossref"}` or `null`.

D.8 Stage 5 (Dependency Resolver): Cross-Reference Correction

Stage 5 (Dependency Resolver): Cross-Reference Correction

Task: Find the correct replacement for a NON-EXISTENT cross-reference.
Input: `{current_id}`, `{current_text}`, `{missing_references}`, `{available_ids}`, `{criteria_with_thresholds}`

Positional Mapping (Roman numerals → Letters): (i)=a, (ii)=b, (iii)=c, (iv)=d, (v)=e, (vi)=f, (vii)=g. So "point X(y)(v)" refers to the 5th child = X(corrected_parent)(e).

Semantic Matching Rules: When "point A(b)(v)" doesn't exist but the correct parent is X(y):
1. Map position: (v) = 5th child = (e)
2. Check if X(y)(e) exists AND its semantic topic matches the context
3. If both match → use X(y)(e)

Example: Text says "comply with point 1(b)(v)" but 1(b)(v) doesn't exist. Missing 1(b)(v) → position (v) = 5th = (e). If correct parent is 1(f), look for 1(f)(e). Read semantic topic of 1(f)(e): "fuel switch by 2035" — if context is about fuel switching → USE 1(f)(e).

Matching Priority:
1. HIGHEST: Criterion with threshold matching the EXACT TOPIC mentioned in text
2. Same threshold type (quantitative values, temporal deadlines)
3. LOWEST: Same structural family (only if no content match found)

Forbidden: Do NOT use parent fallback; do NOT return a parent/ancestor of the current node; do NOT return same ID as both "from" and "to".

D.9 Stage 5 (Dependency Resolver): Threshold Inheritance Detection

Stage 5 (Dependency Resolver): Threshold Inheritance Detection

Task: Detect if criterion inherits thresholds from another criterion.
Input: `{current_id}`, `{current_text}`, `{available_thresholds}`

Chain-of-Thought:
Step 1: Does text have EXPLICIT threshold delegation?
EXPLICIT = [threshold-word] + [delegation-phrase] + [criterion-ID]
- Threshold-words: "threshold", "level", "limit", "value"
- Delegation-phrases: "referred to in", "set out in", "specified in"
- NOT delegation: "criteria", "requirements", "conditions" (too general). "comply with X" without threshold-word (that's a dependency).

Step 2: If explicit delegation found, extract source criterion ID.
Step 3 — Mandatory Selector Detection:
- Period indicators (MUST produce selectors): "over N years", "N-year", "N years average" → `has_period: true, period_value: N`
- Instant indicators: "instant", "at point of generation" → `has_period: false`
- No period mentioned → `threshold_selectors: null` (copy all)

Examples:
`1:` "certifying the level of direct GHG emissions referred to in point X(y)(a)"
Step 1: "level" + "referred to in" + "X(y)(a)" = EXPLICIT. Step 3: no period → null.
→ `{threshold_from: "X(y)(a)", threshold_selectors: null}`
`2:` "comply with the average threshold over 20 years referred to in point A(b)(c)"
Step 1: EXPLICIT. Step 3: "over 20 years" → selector.
→ `{threshold_from: "A(b)(c)", threshold_selectors: [{has_period: true, period_value: 20}]}`
`3:` "Quantified life-cycle GHG emissions are verified by an independent third party"
Step 1: No threshold-word + delegation-phrase + criterion-ID pattern → `{}`
`4:` "Compliance with the criteria set out in this Section is verified"
Step 1: "criteria" is NOT a threshold-word (too general) → `{}`

D.10 Stage 5 (Dependency Resolver): Dependency Detection

Stage 5 (Dependency Resolver): Dependency Detection

Task: Detect applicability dependencies — when one criterion applies ONLY IF another is met/chosen.
Input: `{current_id}`, `{current_text}`, `{available_ids}`

Semantic Principle — Three-way Distinction:
DEPENDENCIES (what we detect): Applicability gating — "This criterion applies ONLY IF criterion X is met/chosen."
NOT dependencies — REFERENCES (Agent 4 handles these): "complies with criteria set out in *Section X*" → points to document to fetch, not conditional applicability.
NOT dependencies — Structural/informational: "as defined in criterion X" → informational, not conditional.

Special Case — Pathway-Specific Anchors: Semantic anchor nodes (`.Verification`, `.BackgroundInformation`) anchored to a SPECIFIC PATHWAY within an OR-alternative group SHOULD have a dependency on that pathway.
Example: `1(f).Verification` is anchored to pathway `1(f)` (one of several OR alternatives under `1`). Verification applies ONLY when activity uses pathway 1(f) → dependency on `1(f)`.
Counter-example: `5.Verification` anchored to top-level `5` (not an OR alternative) → applies universally → NO dependency.

KEY TEST: "Does this criterion apply ONLY to activities that meet another criterion?" YES → Dependency. NO → NOT a dependency.

Detection Steps: (1) Identify conditional language ("where", "if", "when"). (2) Extract the condition. (3) Map condition to criterion ID from available list. (4) Validate it's applicability gating (not structural/informational). (5) Return dependency or null.

Examples:
`1:` "For facilities granted permit by 31 December 2030, capacity does not exceed 15%"
"31 December 2030" is a DATE, not a criterion ID. Inline temporal condition → **null**.
`2:` ID: `5.Verification`. "At beginning of activity, compliance is controlled by authorities"
5 is top-level section (NOT OR-alternative child) → applies universally → **null**.
`3:` ID: `1(f).Verification`. "Where the activity uses pathway 1(f), emissions must be verified"
`1(f)` is child of 1 in OR group → **dependency** on `1(f)`.
`4:` ID: `1(f).BackgroundInformation(a)`. "The Commission shall take those reports into account"
Anchored to pathway `1(f)` (OR child) → **dependency** on `1(f)`.

D.11 Stage 6 (Footnote Processor): Footnote Processing

Stage 6 (Footnote Processor): Footnote Processing

Task: Process regulatory footnote and extract ALL referenced documents.
Input: `{footnote_text}`

Document Types (`kind`): EU Legal Act | International Convention/Treaty | Standard | Official Guideline/Manual | Member-State Plan/Programme | Other.

CELEX ID Construction (EU acts only): Format: `3[YEAR][TYPE][NUMBER]` (10 characters). Type codes: R=Regulation, L=Directive, D=Decision, H=Recommendation.
Year extraction: "YYYY/NNNN" (modern) → first part = year. "NN/YY" (old) → second part = year. IGNORE adoption dates ("of 9 December 1996").
Examples: "Regulation (EU) 2018/1999" → `32018R1999`. "Directive 98/70/EC" → `31998L0070`. "Council Regulation (EC) 338/97" → `31997R0338`.

Type Determination: `must_be_fetched`: needed for compliance ("calculated according to", "complies with"). `citation_only`: background only ("amending", "repealing", "establishing framework").
Splitting Rule: Each legal act is SEPARATE: "amending Regulations 663/2009 and 715/2009, Directives 94/22/EC, 98/70/EC" → 4 items.

Categories (multi-label): **Legal Reference**: ONLY when footnote cites binding legislation. NOT for ISO/EN standards. **Technical Definition**: defines terms. **Official Guideline/Standard**: ISO, IPCC, EN standards. **Explanatory Note**: clarifications, scope context.
Validation: If NO items have kind="EU Legal Act" or "Treaty" → "Legal Reference" MUST NOT be in categories.

D.12 Stage 7 (Schema Assembler): Rule Summary Generation

Stage 7 (Schema Assembler): Rule Summary Generation

Task: Generate a self-contained rule summary (1–2 sentences) using semantic understanding.
Input: `{verbatim_text}`, `{category}`, `{applicability}`, `{tags}`, `{threshold}`, `{references}`

Structure by Classification:
- **Conditional:** Start with condition FIRST ("If...", "Where..."), then state requirement
- **Core Requirement:** WHO must do WHAT — specific and actionable
- **Quantitative:** Embed ALL numeric thresholds with values + units. DO: "must be below 150 g CO_2e/kWh". DON'T: "must meet the threshold". Multiple (OR): include ALL alternatives. Period-based: include time period.
- **Temporal:** Embed dates directly ("by 31 December 2027")
- **Methodology/Verification:** Name specific standard or verifier role

Bracket Notations: Threshold inheritance: append `[THRESHOLD_FROM:ID]`. Cross-reference corrections: append `[CORR FROM:X TO:Y REASON:crossref]`.

Examples:
Quantitative: "Life-cycle GHG emissions must be below 80 g CO_2e/kWh."
Conditional + Methodology: "If the activity blends primary fuels with alternative fuels, the alternative source must comply with sustainability criteria from Directive (EU) 2018/2001: agricultural biomass must meet Article 29(2–5) and forest biomass must meet Article 29(6–7)."
Quantitative + Temporal + Verification: "Direct GHG emissions must be below 150 g CO_2e/kWh or the 15-year average must not exceed 400 kg CO_2e/kW, with compliance verified by 31 December 2027."

D.13 ODR: Observe Step

ODR: Observe Step

Task: Compare the extracted output against the source HTML and enumerate discrepancies.
Input: `{html}`, `{output}`
Issue taxonomy:
- **Structural**: Format violations, missing required fields, malformed IDs
- **Semantic**: Operator inconsistent with text (e.g., ">" when text says "at most")
- **Completeness**: Criteria or thresholds present in source but missing from output
- **Consistency**: Cross-field contradictions (e.g., parent–child category mismatch)

For each issue, report: `{type, severity, field, description, source_evidence}`

D.14 ODR: Diagnose Step

ODR: Diagnose Step

Task: Analyze the observed issues and determine root causes.
Input: `{issues}`, `{output}`, `{history}` (previous attempts)
Produce a DiagnosisResult with:
- **root_cause**: Primary reason for the errors
- **contributing_factors**: Secondary causes
- **recommended_action**: One of {RETRY_MODIFIED, DECOMPOSE, FALLBACK, ACCEPT, ESCALATE}
- **specific_guidance**: Fields to focus on, mistakes to avoid, modifications to apply

D.15 RAG: Query Rewriting with Context

RAG: Criterion-Conditioned Query Rewriting

System: You are an EU regulatory retrieval expert. Reformulate EU Taxonomy criterion text into a natural-language question optimized for finding relevant passages in EU regulatory documents.
Task: Reformulate the criterion below into a retrieval-optimized question incorporating activity context.
Input: `{query}`, `{activity_name}`, `{objective}`, `{criteria_section}`, `{article_ref}`

Instructions:
1. EXPAND all abbreviations/acronyms to full EU regulatory forms
2. FRAME as: "For the [substantial contribution / DNSH] criteria of [activity] under [objective], what does [regulation] require regarding [topic]?"
3. INCLUDE the article/section reference naturally if provided
4. INCLUDE key regulatory concepts: requirements, thresholds, definitions, conditions
5. KEEP numeric values, units, and article references exact
6. NEVER invent references: only mention Articles/Appendices/Annexes that appear in the criterion text

Examples:
Activity: Electricity generation using solar PV. Objective: mitigation. Section: substantial contribution.
Criterion: "The activity complies with the emission threshold set in Article 29(4)(a)"
→ "For the substantial contribution of electricity generation using solar photovoltaic technology to climate change mitigation, what emission threshold does Article 29, paragraph 4, points (a) establish, including the specific CO_2 limit values and measurement conditions?"
Activity: Afforestation. Objective: adaptation. Section: DNSH.
Criterion: "prevention and avoidance of introduction of invasive alien species"
→ "For the DNSH criteria of afforestation under climate change adaptation, what measures does EU regulation require to prevent the introduction and spread of invasive alien species, including species of Union concern, containment conditions, and risk assessment?"

D.16 RAG: Retrieval Evaluation

RAG: Retrieval Relevance Evaluation

System: You are an EU regulatory document analysis expert. Evaluate whether passages contain the substantive requirements relevant to the criterion. Focus on CONTENT — the actual rules, thresholds, definitions, and conditions — not on whether a specific article number appears verbatim.
Task: Evaluate whether retrieved passages contain the substantive content needed.
Input: `{criterion}`, `{celex_id}`, `{passages}`

Scoring rubric:
- 0.9–1.0: Actual regulatory requirements, thresholds, definitions, or conditions
- 0.7–0.89: Related requirements but missing specific details
- 0.5–0.69: Correct regulation but different topic
- Below 0.5: Unrelated

Do NOT penalize passages for lacking a specific article/section heading. A passage containing the actual emission threshold or definition is highly relevant even without an "Article X" heading.
Output: `{confidence, relevant_passages, gaps, reasoning}`

D.17 RAG: Query Refinement

RAG: Iterative Query Refinement

System: You are a search query generator. Output ONLY a single search query sentence. No reasoning, no explanation, no first person, no preamble.
Task: Given the original query and retrieval gaps, generate a better, more specific search query.
Input: `{original_query}`, `{gaps}`
Focus on the specific legal terms, article numbers, or thresholds mentioned in the gaps.

D.18 RAG: Safe Summarization

RAG: Safe Summarization

System: You are an EU regulatory compliance expert. Summarize relevant passages from EU legal documents in the context of a specific Technical Screening Criterion.
Rules:
1. Focus on requirements, thresholds, dates, and conditions relevant to the criterion
2. Quote key thresholds and dates VERBATIM (in quotation marks)
3. Keep each direct quote under 100 words
4. Identify specific articles/sections that apply
5. Note any conditions or exceptions
6. Be factual — do NOT invent requirements not in the source text

Input: `{criterion}`, `{celex_id}`, `{passages}`
Output: `{text, key_facts, thresholds, confidence}`

D.19 Semantic Equivalence Judge Prompts

The following prompt templates are used by the GPT-4o LLM judge (max 150 tokens) to score semantic equivalence between system-extracted

and gold-annotated fields. Each prompt receives three inputs: the *verbatim regulatory text* (ground truth source), the *gold* (human-annotated) field, and the *system* (pipeline-extracted) field. Scores range from 0 to 5, where **0** indicates concordant absence (both sides are null/empty, automatically assigned without an LLM call) and **1–5** follow the rubric below. Reported statistics in Table 2 are computed over scores > 0 only, i.e., on the 1–5 scale. All four prompts share the same structure: schema definition, numbered evaluation rules with point deductions, scoring rubric with concrete examples, and JSON output format. We present the threshold prompt in full and abbreviate the remaining three to their schema and rubric sections; evaluation rules follow the same pattern of field-level matching with graduated penalties.

Threshold Equivalence Judge (full prompt)

You are an expert evaluator for EU Taxonomy regulatory criteria extraction. Compare the SYSTEM-extracted threshold against the GOLD (human-annotated) threshold for a single criterion node.

Schema. Each threshold object has two sub-fields:
- `quantitative`: {logic: AND|OR|null, items: [{metric, operator, value, unit, period}]} or null
- `temporal`: {logic: AND|OR|null, items: [{type: deadline|effective_from|window|recurring, ...}]} or null

Evaluation Rules:
1. *Structural placement is irrelevant*: recurring intervals under "temporal" vs. "quantitative" are not penalized if semantic content is identical.
2. *Metric label is free-text*: "Primary Energy Demand" vs. "PED" are equivalent; judge whether they refer to the same physical quantity.
3. *Operator equivalence*: $\geq$ and "at least" are the same; $>$ vs. $\geq$ *is* a meaningful difference (-1 pt).
4. *Value must be numerically identical*: 10 vs. 10.0 is fine; 10 vs. 100 is wrong.
5. *Unit normalization*: "%" vs. "percent", "g CO2e/kWh" vs. "gCO2eq/kWh" are equivalent; "g" vs. "kg" is wrong.
6. *Logic field*: AND vs. OR with $\geq$ 2 items is a critical error (-2 pts). With 1 item, logic=null is expected.
7. *Period field*: missing but inferable from context is minor (-1 pt); wrong value or unit is major (-2 pts).
8. *One-sided presence*: if one side is null, check verbatim text for an explicit numeric constraint.

Inputs: `{verbatim}`, `{gold}`, `{system}`

Scoring Rubric (0–5):
- **5 — Full equivalence**: All items match on value, operator, unit, metric concept, and temporal constraints. Surface differences in label wording or sub-field placement are acceptable.
- **4 — One minor field-level error**: All items present with correct value/operator/unit, but exactly one minor issue (vague metric label, missing period when inferable, null logic with $\geq$ 2 correct items).
- **3 — Partial capture, one substantive gap**: Primary constraint correct, but one item entirely missing from a multi-item set, or operator direction wrong on one item, or temporal condition absent.
- **2 — Significant errors**: Threshold recognized but critical details wrong (wrong numeric value, fundamentally different unit, only 1 of 3+ items, AND/OR inverted).
- **1 — Wrong or unjustified**: False positive (extraction where text is qualitative) or false negative (null where text has explicit limits).
- **0 — Not applicable**: Both sides null/empty; concordant absence.

Output: `{"score": <int 0-5>, "reason": "<one sentence>"}`

Reference Equivalence Judge (abbreviated)

Schema. Each references object has: `logic` (AND|OR|null), `sources`: [{text, type: must_be_fetched|citation_only, link, link_status, celex_id}].

Key Evaluation Rules: Sources are matched by CELEX ID, directive/regulation number, or standard identifier (not by text wording). Type mismatches (must_be_fetched vs. citation_only) incur -1 pt each. Missing sources incur -1 pt each. Extra sources present in the verbatim text are acceptable; those absent from the text are false positives (-1 pt). Logic field AND vs. OR with $\geq$ 2 sources: -1 pt if wrong.

Scoring Rubric (0–5):
- **5**: Same set of legal acts/standards, correct type classification, correct logic.
- **4**: All sources identified; exactly one minor discrepancy (one type mismatch, or null logic with $\geq$ 2 sources).
- **3**: One source missing or one false positive.
- **2**: Two or more sources missing, or >50% type mismatches.
- **1**: Most gold sources absent, or entirely wrong legal acts cited.
- **0**: Both sides null/empty; concordant absence.

Footnote Equivalence Judge (abbreviated)

Schema. Each footnote has: `id`, `verbatim` (primary content from HTML popover), `categories` (Legal Reference | Technical Definition | Official Guideline/Standard | Explanatory Note), `items` (structured references with kind, celex_id, type), `definitions` [{term, definition}], `notes`.

Key Evaluation Rules: Footnotes are matched by *content* (verbatim field), not by ID. Minor whitespace/encoding differences are acceptable; truncation (>20% missing) is substantive. Category mismatches: extra secondary label -0.5 pt, wrong primary category -1 pt. Missing parsed items when footnote cites legal acts: -1 pt. Missing definitions when footnote defines a term: -1 pt.

Scoring Rubric (0–5):
- **5**: Same footnote set (by content); categories,

items, definitions all agree.
- **4**: All footnotes present; exactly one minor field error (extra category label, wrong item kind, slightly reworded definition).
- **3**: One footnote missing, or one with truncated content, or items/definitions array empty when clearly needed.
- **2**: Two or more footnotes missing, or fundamentally different verbatim content.
- **1**: Most footnotes missing or wrong content; would mislead a user.
- **0**: Both sides have empty lists; concordant absence.

Dependency Equivalence Judge (abbreviated)

Schema. Each dependencies object has: `condition_summary` (plain-language description), `min_conditions_to_meet` (int; 1=OR, len(clauses)=AND), `clauses`: [{criterion_id, status: Affirmation|Negation}]. "Affirmation" = target must be met; "Negation" = target must NOT be met.

Key Evaluation Rules: Clauses matched by criterion_id (minor format variations acceptable: "2.1" = "2.1."). Status mismatch (Affirmation vs. Negation) is a major error (-2 pts) as it inverts dependency logic. min_conditions_to_meet wrong with ≥ 2 clauses: -2 pts (changes AND/OR semantics). condition_summary judged by logical meaning, not wording; wrong criterion reference: -1 pt.

Scoring Rubric (0–5):
- **5**: All clauses match on criterion_id, status, and min_conditions_to_meet; condition_summary conveys same meaning.
- **4**: Correct structure; exactly one minor discrepancy (vague summary, minor ID format difference, min_conditions differs with only 1 clause).
- **3**: Primary dependency correct; one secondary clause missing, or min_conditions wrong with ≥ 2 clauses.
- **2**: Dependency recognized but key error: status inverted, or criterion_id points to wrong node.
- **1**: False negative (null when text has conditional language) or false positive (fabricated dependencies).
- **0**: Both sides null; concordant absence.

D.20 RAG Quality Judge Prompts

The following prompt templates are used by the GPT-4o judge (max 200 tokens) to score RAG summary quality. Each prompt receives four inputs: the *regulatory criterion* being enriched, the *retrieved source passages* from the referenced CELEX document, the *generated summary*, and the *extracted key facts and thresholds*. Scores range from 1 to 5 following a graduated rubric; the judge returns a JSON object with a score and one-sentence justification.

Faithfulness Judge

You are an expert evaluator for regulatory RAG pipeline outputs. Assess whether the SUMMARY faithfully represents the SOURCE PASSAGES.

Inputs: `{criterion_text}`, `{chunk_texts}`, `{summary_text}`, `{key_facts}`, `{thresholds}`.

Evaluation Rules:
1. Faithfulness = grounding. Every claim in the summary must be traceable to the source passages. A claim that is true in general EU law but not in these specific passages is hallucination.
2. Numeric thresholds in the summary or thresholds list must appear (exactly or equivalently) in the source passages.
3. Each key fact must be supported by at least one passage.
4. Omission of passage content is acceptable—only fabrication is penalized.
5. Article/section references cited in the summary must appear in the passages.

Scoring Rubric (1–5):
- **5 — Fully faithful**: Every claim, threshold, and key fact is directly supported by the passages. Paraphrasing is acceptable if meaning is preserved.
- **4 — One minor unsupported detail**: All major claims are grounded. Exactly one minor detail lacks direct support: a slightly overstated paraphrase, a vague article reference, or a key fact that combines two passage ideas in a way not explicitly stated.
- **3 — One substantive unsupported claim**: Core summary is grounded, but one substantive claim is not supported: a threshold value not in the passages, a key fact that conflates distinct requirements, or a specific regulatory reference not found in the retrieved text.
- **2 — Multiple unsupported claims**: Two or more claims lack passage support, or a critical threshold is hallucinated. The summary would mislead a compliance assessor on material points.
- **1 — Mostly hallucinated**: The majority of summary content cannot be traced to the passages. The summary appears drawn from general knowledge rather than retrieved text.

Output: `{"score": <int 1-5>, "reason": "<one sentence>"}`

Relevance Judge

You are an expert evaluator for regulatory RAG pipeline outputs. Assess whether the SUMMARY is relevant to the regulatory CRITERION that triggered the retrieval.

Inputs: `{criterion_text}`, `{chunk_texts}`, `{summary_text}`, `{key_facts}`, `{thresholds}`.

Evaluation Rules:
1. Classify each summary claim into three tiers: *directly relevant* (addresses a specific requirement of the criterion), *contextual* (provides useful background but does not directly address the criterion), or *unrelated* (no connection to the criterion).
2. The criterion text defines relevance—not the passages. A claim may be faithful to the passages but irrelevant to the criterion.

3. General legal context is "contextual" unless it directly clarifies a requirement in the criterion.
4. Key facts should each relate to a specific aspect of the criterion.

Scoring Rubric (1–5):
- **5 — Fully relevant**: Every claim in the summary directly addresses a requirement or condition stated in the criterion. Key facts map to specific criterion aspects.
- **4 — One contextual claim**: All major claims are directly relevant. Exactly one claim provides background context rather than addressing the criterion directly.
- **3 — Multiple contextual or one unrelated**: Several claims provide only background context, or one claim is entirely unrelated to the criterion.
- **2 — Significant off-topic content**: The summary contains substantial content unrelated to the criterion, diluting the useful information.
- **1 — Wrong topic**: The summary addresses a different regulatory topic than the criterion.

Output: `{"score": <int 1-5>, "reason": "<one sentence>"}`

Completeness Judge

You are an expert evaluator for regulatory RAG pipeline outputs. Assess whether the SUMMARY captures all compliance-critical information from the SOURCE PASSAGES.

Inputs: `{criterion_text}`, `{chunk_texts}`, `{summary_text}`, `{key_facts}`, `{thresholds}`.

Evaluation Rules:
1. Identify all compliance-critical elements in the passages: quantitative thresholds, mandatory requirements, conditions, deadlines, and exceptions.
2. Classify each element as *major* (quantitative thresholds, mandatory requirements, binding conditions) or *minor* (definitions, contextual details, non-binding guidance).
3. Check whether each element is captured in the summary or key facts/thresholds lists.
4. A major element omitted from all three outputs (summary, key facts, thresholds) is a substantive gap.

Scoring Rubric (1–5):
- **5 — Fully complete**: All major and minor compliance-critical elements from the passages are captured in the summary, key facts, or thresholds.
- **4 — One minor omission**: All major elements captured. Exactly one minor element (a definition, a non-binding recommendation) is omitted.
- **3 — One major omission**: One major compliance-critical element is missing, or multiple minor elements are omitted.
- **2 — Multiple major omissions**: Two or more major elements are missing. A compliance assessor would lack critical information.
- **1 — Mostly incomplete**: The majority of compliance-critical content from the passages is missing from the summary.

Output: `{"score": <int 1-5>, "reason": "<one sentence>"}`

Table 6: Stages of the REGREACT pipeline. Each stage receives the cumulative extraction state and the original HTML, applying its specialized competency before passing control to the next stage.

Stage	Role	Primary Output
s_1	Structural Parser	Criterion hierarchy from HTML
s_2	Threshold Extractor	Quantitative & temporal thresholds
s_3	Content Classifier	Category, applicability, eval. logic
s_4	Reference Extractor	External sources & CELEX IDs
s_5	Dependency Resolver	Inter-criteria relationships
s_6	Footnote Processor	Footnote text & anchor linkage
s_7	Schema Assembler	Validated JSON output

Coverage Judge

You are an expert evaluator for regulatory RAG pipeline outputs. Assess whether the SUMMARY addresses the information needs expressed by the CRITERION.

Inputs: `{criterion_text}`, `{chunk_texts}`, `{summary_text}`, `{key_facts}`, `{thresholds}`.

Evaluation Rules:
1. Analyze the criterion to identify its information needs: what specific regulatory content does the criterion reference or require?
2. Classify each need as *primary* (the specific requirement, threshold, or condition the criterion explicitly references) or *secondary* (supporting context that aids interpretation).
3. Coverage measures retrieval adequacy: did the pipeline retrieve and summarize passages that address what the criterion asks for?
4. A criterion referencing a specific article expects the summary to cover that article's content. A criterion referencing a general regulation expects broader coverage of relevant provisions.

Scoring Rubric (1–5):
- **5 — Full coverage**: All primary and secondary information needs of the criterion are addressed by the summary.
- **4 — One secondary need unmet**: All primary needs addressed. One secondary information need (supporting context, related provision) is not covered.
- **3 — Partial primary coverage**: One primary need is only partially addressed, or multiple secondary needs are unmet.
- **2 — Inadequate coverage**: The primary information need is only partially addressed. A compliance assessor would need to consult additional sources.
- **1 — No coverage**: The summary does not address the criterion's core information need.

Output: `{"score": <int 1-5>, "reason": "<one sentence>"}`

E Supplementary Methodology Details

E.1 Semantic Anchoring Procedure

For unnumbered paragraphs that are not resolved by the structural fast-path (§3.3), the LLM performs a three-stage analysis:

1. **Reference extraction**: the LLM identifies all criterion references within the paragraph (e.g., "point 1(f)", "criteria referred to in 1(b)") with confidence scores;
2. **Scope analysis**: determines whether the paragraph applies to a single criterion, a range, or an entire section, using the extracted references and surrounding context;
3. **Hierarchy determination**: assigns the paragraph's position in the hierarchy (sibling vs. child of the referenced criterion) based on its semantic function.

A separate *semantic classification* step then applies a three-part reasoning chain (*audit testability*, *obligation direction*, and *compliance impact*) to assign each paragraph a descriptive type such as Verification, Assessment, or BackgroundInformation. This type is used to generate stable, semantically meaningful identifiers (e.g., `1(f).Verification`) with deduplication.

Data Acquisition. The technical screening criteria for all 242 activities were obtained from the EU Taxonomy Compass,[2] the European Commission's official platform for navigating the Taxonomy Regulation. The platform provides each activity's criteria as raw HTML; all extraction challenges described in §1—implicit hierarchy, unnumbered paragraphs, ambiguous cross-references—are fully present in this input.

Baseline Selection. We compare against a GPT-4o single-pass baseline rather than prior systems (Table 1) for three reasons: (1) each system targets different regulatory texts with incompatible output schemas and extraction granularity, making score comparison meaningless; (2) adapting a system designed for a different regulatory domain to the EU Taxonomy would require reimplementing its extraction logic, risking an uncharitable strawman comparison; and (3) our baseline uses GPT-4o, a frontier commercial LLM, which receives the identical input HTML, extraction specification, output schema, and few-shot examples in a single call, providing a controlled test of whether multi-agent decomposition with a smaller model outperforms monolithic generation with a stronger one.

Ablation Study. To isolate the contribution of each mechanism, we evaluate two ablated variants: **−ODR**, which removes all self-correction loops so each stage runs once; and **−Graph**, which disables cross-stage structural validation (Table 7).

[2] https://ec.europa.eu/sustainable-finance-taxonomy/taxonomy-compass

Table 7: Ablation results on n=100 gold-annotated activities. Structural and classification metrics are percentages; semantic equivalence uses a 1–5 GPT-4o judge scale.

Metric	GPT-4o	−ODR	−Graph	Full
Structural & Classification (%)				
Structural F1	78.6	89.5	90.8	**94.12**
Category Acc.	90.2	96.3	98.1	**98.6**
Applicability Acc.	85.7	93.6	96.4	**97.24**
Eval Logic Acc.	80.3	89.2	87.6	**93.4**
Semantic Equivalence (1–5)				
Threshold	3.23	4.12	4.35	**4.43**
Reference	3.34	4.38	4.71	**4.77**
Footnote	3.12	4.09	4.39	**4.48**
Dependency	2.96	4.21	4.35	**4.63**

Removing ODR retains reasonable structural quality (89.5% F1) thanks to graph enforcement, but degrades semantic metrics, with Dependency showing the largest drop (4.63→4.21) as cross-reference corrections (§3.3, Stage 5) require iterative analysis across multiple passes. Removing the graph preserves semantic content (Reference 4.71, Threshold 4.35) but causes the largest structural drop on Evaluation Logic (93.4→87.6%), since the graph enforces logic–child-count consistency (§3.5) that no single stage can verify. Both components contribute complementary gains: ODR refines within-stage semantic accuracy, while the graph ensures cross-stage structural coherence.

E.2 CELEX Identifier Parsing

EU legislation references are normalized to CELEX identifiers through a four-step parsing chain: (1) extract year and document number via format-specific patterns; (2) determine document type from contextual keywords (Regulation, Directive, Decision); (3) construct the machine-readable identifier (e.g., `32018R1999`); (4) validate format and recover from common malformations (transposed digits, missing prefixes).

E.3 Structural F1 Computation

Structural F1 measures how well a system-produced criterion tree matches the gold-standard tree. The computation proceeds in two stages.

Node Alignment. Both trees are flattened into node lists, and nodes are paired through a two-pass process. The first pass matches nodes by exact `criterion_id`. The second pass takes all remaining unmatched nodes and pairs them greedily by

verbatim text similarity, accepting any pair whose `SequenceMatcher` ratio reaches at least 0.80. This produces three sets: matched pairs, unmatched gold nodes (false negatives), and unmatched system nodes (false positives).

Context Quality Score. Each matched pair receives a quality score between 0 and 1, computed as a weighted sum of four features:

1. **Parent placement** (weight 0.30): whether the system node sits under the same parent `criterion_id` as the gold node. A correctly identified node placed under the wrong parent receives no credit for this feature.
2. **Sibling overlap** (weight 0.25): Jaccard similarity between the sibling ID sets of the system and gold nodes. If gold groups criteria {a, b, c} under one parent but the system groups {a, b, d}, partial credit reflects the overlap.
3. **Subtree shape** (weight 0.25): ratio of child counts, computed as $\min(s, g)/\max(s, g)$ where s and g are the number of children in the system and gold trees. A leaf node matched to another leaf receives full credit.
4. **Schema completeness** (weight 0.20): fraction of the 13 required output fields present as keys on the system node. A node missing two of thirteen fields receives $11/13 \approx 0.85$ for this feature.

F1 Computation. The quality scores are summed across all matched pairs to produce a weighted true-positive count. Precision divides this count by the total number of system nodes (matched plus unmatched), and recall divides it by the total number of gold nodes (matched plus unmatched). F1 is the harmonic mean of precision and recall.

E.4 RAG Query Refinement

Figure 8 illustrates the ReAct loop's query refinement across iterations. Queries become progressively more specific, targeting particular articles and provisions.

F Output Schema

Each criterion node in the EU-TAXOSTRUCT dataset conforms to the following fixed 13-field JSON schema. Complex objects (`threshold`, `references`, `dependencies`) are set to `null` when they contain no meaningful data; `footnotes` is always an array (`[]` when empty, never `null`).

Query Refinement Examples (3-Iteration Cases)

Example 1: Biodiversity — Habitats Directive (92/43/EEC)
Final retrieval confidence: **0.90**

Iteration 1: "What are the conditions for classification of protected areas under the IUCN system and Natura 2000 sites according to Directive 92/43/EEC?"

Iteration 3: "What specific legal provisions does Directive 92/43/EEC establish for classifying protected areas under the EU Taxonomy?"

Example 2: Afforestation — EU Taxonomy (2020/852)
Final retrieval confidence: **0.48**

Iteration 1: "What does Article 11(1) of the EU Taxonomy define as substantial contribution for afforestation under climate change adaptation?"

Iteration 3: "What are the specific criteria and thresholds under Article 11(1) for demonstrating substantial contribution in afforestation?"

Figure 8: Query refinement across ReAct iterations. Example 1 achieves high confidence after narrowing scope to specific provisions. Example 2 correctly assigns low confidence when the target article lacks retrievable thresholds.

Fixed 13-Field Criterion Schema

```
{
  "criterion_id": "",
  "category": "Quantitative" | "Qualitative",
  "applicability": "Mandatory" | "Conditional",
  "tags": ["Verification", "Methodology", "Commitment",
           "Assessment", "Replacement",
           "BackgroundInformation"],
  "verbatim_text": "",
  "rule_summary": "",
  "evaluation_logic": "AND" | "OR" | "N_OF_K" | "LEAF",
  "n_required": null,
  "threshold": null | {
    "quantitative": null | {
      "logic": "AND" | "OR" | null,
      "items": [
        {
          "metric": "",
          "operator": ">" | "<" | ">=" | "<=" | "=",
          "value": 0,
          "unit": "",
          "period": null | {
            "type": "bounded",
            "value": 0,
            "unit": "years" | "months" | "days"
          } | {
            "type": "lifetime"
          }
        }
      ]
    },
    "temporal": null | {
      "logic": "AND" | "OR" | null,
      "items": [
        { "type": "deadline", "date": "YYYY-MM-DD" } |
        { "type": "effective_from",
          "date": "YYYY-MM-DD" } |
        { "type": "effective_until",
          "date": "YYYY-MM-DD" } |
        { "type": "window",
          "start": "YYYY-MM-DD",
          "end": "YYYY-MM-DD" } |
        { "type": "recurring",
          "interval_value": 0,
          "interval_unit": "years" | "months" | "days" }
      ]
    }
  },
  "references": null | {
    "logic": "AND" | "OR" | null,
    "sources": [
      {
        "text": "",
        "type": "citation_only" | "must_be_fetched",
        "link": "",
        "link_status": "provided" | "generated",
        "celex_id": null | "32018L2001",
        "enrichment": null | {
          "status": "retrieved"|"paywalled"|"skipped",
          "summary": "",
          "key_facts": [],
          "thresholds": [],
          "confidence": 0.0
        }
      }
    ]
  },
  "dependencies": null | {
    "condition_summary": "",
    "min_conditions_to_meet": 1,
    "clauses": [
      { "criterion_id": "", "status": "Affirmation" | "Negation" }
    ]
  },
  "sub_criteria": [],
  "footnotes": [
    {
      "id": "fn-1",
```

```
      "categories": [],
      "items": [
        {
          "kind": "EU Legal Act" | "Standard" | "...",
          "title": "",
          "celex_id": null | "32018L2001",
          "type": "citation_only" | "must_be_fetched",
          "oj": null | "",
          "enrichment": null | {
            "status": "retrieved"|"paywalled"|"skipped",
            "summary": "",
            "key_facts": [],
            "thresholds": [],
            "confidence": 0.0
          }
        }
      ],
      "definitions": [],
      "notes": []
    }
  ]
}
```

Figure 9: Fixed 13-field JSON schema for each criterion node in EU-TAXOSTRUCT. The `enrichment` sub-object inside each reference source and footnote item is populated by the RAG sub-pipeline (§3.6); fields `summary` through `confidence` are present only when `status` = `"retrieved"`. See Figure 4 for a populated example.

G Gold Annotation Details

The 100 gold-annotated activities were selected via stratified sampling across three structural complexity tiers:

- **Simple** (30 activities): leaf-node criteria with a single threshold and no inter-criteria dependencies.
- **Medium** (40 activities): a chapeau node with 2–5 children connected by basic AND/OR evaluation logic.
- **Complex** (30 activities): transitional pathways, ancillary nodes, footnotes, cross-reference corrections, and threshold inheritance.

Activities were balanced across all three Delegated Acts to ensure representation of different drafting conventions.

The extraction pipeline generated candidate outputs for each activity. Two PhD students, both trained on the extraction specification (Appendix F), served as annotators. Each annotator independently reviewed 50 activities across nine dimensions: applicability, category, evaluation logic, tags, thresholds, references, dependencies, footnotes, and structure. The two sets were assigned so that each complexity tier and each Delegated Act were represented in both.

To measure annotation reliability, a shared subset of 15 activities (5 per complexity tier) was independently annotated by both. Table 8 reports Cohen's κ for the three core evaluation dimensions.

Table 8: Inter-annotator agreement on the shared 15-activity subset.

Dimension	Cohen's κ
Hierarchical structure	0.91
Dependencies (incl. cross-reference corrections)	0.86
Threshold extraction	0.84

Disagreements on the shared subset were resolved through discussion until consensus was reached. The resulting corrections were applied to produce the final gold annotations used throughout the evaluation.

Judge Validation. To validate LLM-judge reliability, one annotator independently re-scored a stratified random subset of 50 criterion–dimension pairs using the same rubrics and 1–5 scales: 30 pairs drawn from both REGREACT and baseline outputs across the four semantic equivalence dimensions, and 20 pairs from REGREACT outputs across the four RAG quality dimensions. Human–judge agreement yielded Spearman ρ=0.86 overall (0.87 for semantic equivalence, 0.85 for RAG quality), with 91% of scores falling within ± 1 point and Cohen's κ=0.83 on binarized ratings (≤ 3 vs. ≥ 4).

H Extended Evaluation

H.1 Task Definition

The task is formalised in §3.1: each activity in a regulatory document is supplied as a semi-structured HTML fragment h_i and must be mapped to a structured representation $\mathcal{S}_i = (\mathcal{C}_i, \phi_i, \mathcal{T}_i, \mathcal{R}_i, \mathcal{G}_i, \xi_i)$ encoding the hierarchical criterion tree, per-node evaluation logic, thresholds, external references and inter-criteria dependencies, the typed criterion graph, and resolved reference summaries. Each criterion node conforms to the 13-field schema in Appendix F. Correctness is evaluated structurally—by `criterion_id` alignment and parent placement against gold trees (Appendix E.3)—and semantically, on a 0–5 graduated rubric in which 0 marks concordant absence and 1–5 range from wrong to full equivalence (§5, Appendix D.19). The *self-containedness* property requires that every external reference and footnote dependency be resolved and embedded inline, so that the output needs no further document consultation.

H.2 Self-Refine Baseline

We evaluate a Self-Refine baseline (Madaan et al., 2023) using GPT-4o on the same n=100 gold-annotated activities. The baseline first produces a full extraction; GPT-4o then critiques its own output against the source HTML and rewrites it, repeating this critique-and-revise cycle up to three times and stopping earlier if the critique finds no issues. Unlike REGREACT, it operates on the full output at once rather than on each specialised stage, and it has no typed graph to enforce structural consistency between iterations. Table 9 reports its performance alongside the gaps against the single-pass and REGREACT numbers from Table 3.

Table 9: Self-Refine performance and gaps (n=100). Single-pass and REGREACT scores appear in Table 3. Structural metrics in %; semantic equivalence on the 1–5 scale.

Metric	Self-Refine	Δ vs Single-pass	Δ vs REGREACT
Structural F1	86.3	+7.7	−7.8
Evaluation Logic Acc.	85.1	+4.8	−8.3
Threshold	3.85	+0.62	−0.58
Dependency	3.64	+0.68	−0.99

Self-Refine closes between roughly one third and

one half of the gap on every metric, confirming that iterative refinement with access to the source provides genuine benefit. The largest residual gap is on Evaluation Logic Accuracy (-8.3, only 37% of the gap closed), where the typed graph enforces logic-versus-child-count consistency (§3.5) that free-form critique cannot guarantee, and on Dependency (-0.99), where Stage 5 resolves cross-reference corrections through dedicated semantic analysis that monolithic critique misses entirely.

H.3 Computational Cost

Each REGREACT stage runs an initial extraction call followed by an observe call comparing the output against the source (§3.4); when issues are detected, up to two correction cycles (diagnose + repair + observe) are appended, yielding 2, 5, or 8 calls per stage depending on correction depth. The RAG sub-pipeline processes a mean of 2.93 enriched reference pairs per activity (709 pairs across 242 activities); Table 5 reports 75.4%, 12.4%, and 12.1% of pairs resolving in one, two, or three retrieval iterations respectively, yielding a mean of 3.7 calls per pair and approximately 10.9 RAG calls per activity. We price GPT-4o at \$2.50 per million input tokens and \$10.00 per million output tokens, and DeepSeek-R1-Distill-Qwen-32B at \$0.27 per million tokens for both input and output via the DeepInfra endpoint—used here as an API-equivalent reproducibility benchmark for our self-hosted deployment. Table 10 reports the resulting mean cost per activity.

Table 10: Mean estimated cost per activity (n=100).

System	Calls	Input tok.	Output tok.	USD
Single-pass (GPT-4o)	1.0	4,500	2,000	0.031
Self-Refine (GPT-4o)	3.4	18,100	4,250	0.088
REGREACT (32B)	32.6	85,900	17,670	0.028

Despite issuing roughly 30× more LLM calls than single-pass, REGREACT operates at a slightly *lower* dollar cost than the single-pass GPT-4o baseline and at approximately one third the cost of Self-Refine. The savings stem from the 32B backbone, which is priced symmetrically and avoids GPT-4o's \$10.00 per million output tokens—the dominant cost driver for both monolithic baselines.

H.4 Cross-Model Judge Validation

To check that the semantic-equivalence gaps in Table 3 are not artefacts of GPT-4o judging outputs from a related model family, we re-evaluate the four semantic-equivalence dimensions on a stratified subset of 50 activities using Claude Sonnet 4.5 as an independent judge with the identical rubrics (Appendix D.19). Across all four dimensions, the gap between REGREACT and the single-pass baseline measured by Claude remains within 0.15 points of the gap measured by GPT-4o, preserving the ranking in Table 3. Table 11 reports the per-dimension agreement between the two judges.

Table 11: Agreement between the GPT-4o and Claude Sonnet 4.5 judges on 50 stratified activities. Spearman ρ measures whether the two judges rank criterion–source pairs in the same order; "% within ±1" is the fraction of pairs whose scores differ by at most one point on the 1–5 scale.

Dimension	Spearman ρ	% within ±1
Threshold	0.87	92
Reference	0.89	94
Footnote	0.84	89
Dependency	0.85	90

Spearman ρ exceeds 0.84 on every dimension and at least 89% of paired scores fall within one point, with the strongest agreement on Reference—where CELEX normalisation gives the judges a discrete, verifiable target—and the weakest on Footnote, where multi-item categorisation leaves more interpretive room. The conclusions in §6 are therefore robust to judge choice.

H.5 Cross-Reference Correction Evidence

Cross-reference corrections, the CORRECTS edges produced by Stage 5, are validated against human annotation rather than against the system's own self-assessment. The 100 gold-annotated activities were independently reviewed by two PhD students following the extraction specification in Appendix F, with Cohen's $\kappa = 0.86$ on the dependency dimension—which includes cross-reference corrections—on the shared 15-activity subset (Table 8); disagreements were resolved through discussion before producing the final gold. The Dependency equivalence score of 4.63 (Table 2) and its $+1.67$ gap over the single-pass baseline (Table 3) are therefore judged against human-validated corrections, not against the pipeline's own claims about what the regulation intends.

H.6 Rationale for Seven Stages

The seven stages decompose extraction by linguistic and structural competency. Stages 1 and 7 address *structural hallucination* (§1, failure mode i)

through parsing and assembly; Stages 2, 3, and 6 handle the heterogeneous nature of regulatory content via numeric thresholds, logical classification, and footnotes respectively; and Stages 4 and 5 address *reference incompleteness* (failure mode iv) by extracting and resolving external and inter-criteria dependencies. The remaining failure modes, *cross-stage inconsistency* (ii) and *structural violation* (iii), are handled by cross-cutting mechanisms rather than individual stages: the shared semantic memory (§3.2) maintains consistency across stages, and the typed criterion graph (§3.5) enforces structural invariants. A literal per-stage ablation is not well defined for a sequential pipeline, since removing the structural parser leaves later stages with no input. Table 7 therefore reports the principled decomposition along the two cross-cutting mechanisms: removing ODR drops Structural F1 by 4.62 points (94.12 $\rightarrow$ 89.5), and removing the typed graph drops Evaluation Logic Accuracy by 5.8 points (93.4 $\rightarrow$ 87.6). Both mechanisms contribute distinct and measurable shares of the overall gain.