

COMPACT: Building Compliance Paralegals via Clause Graph Reasoning over Contracts

Ayush Singh* Dishank Aggarwal* Pranav Bhagat*
Ainulla Khan Sameer Malik Amar Prakash Azad

Fujitsu Research of India

{ayush.singh, dishank.aggarwal, pranav.bhagat,
ainulla.khan, sameer.malik, amar.azad}@fujitsu.com

Abstract

Contract compliance verification requires reasoning about cross-clause dependencies where obligations, exceptions, and conditions interact across multiple provisions, yet existing legal NLP benchmarks like ContractNLI and CUAD focus exclusively on isolated single-clause tasks. We introduce COMPACT (COMpliance PARalegals via Clause graph reasoning over conTracts), a framework that models cross-clause dependencies through structured clause graphs. Our approach extracts deontic-temporal entities from clauses and constructs typed relationship graphs capturing definitional dependencies, exception hierarchies, and temporal sequences. From these graphs, we introduce ACE (Assessing Compliance in Enterprise)— a benchmark containing 4,700 carefully constructed compliance scenarios derived from 633 real-world contracts covering 26 types of agreements. Each scenario requires multi-hop reasoning across multiple clauses, and undergoes independent LLM-based validation to ensure quality. Evaluation reveals that multi-clause reasoning poses a fundamental challenge for state-of-the-art models (34–57% base accuracy), while training on ACE yields substantial improvements on compliance tasks (+22–43 % points) and also enhances general legal reasoning performance on other benchmarks (PrivaCI-Bench, ContractNLI).

1 Introduction

Legal compliance is a critical requirement across industries such as finance, healthcare, technology, and manufacturing. Non-compliance can result in costly disputes, regulatory penalties, and reputational damage (Susskind and Susskind, 2017). To mitigate these risks, enterprises rely on lawyers who manually review contracts clause-by-clause to identify obligations, exceptions, and potential conflicts. However, this process is slow, expensive,

*These authors contributed equally to this work.

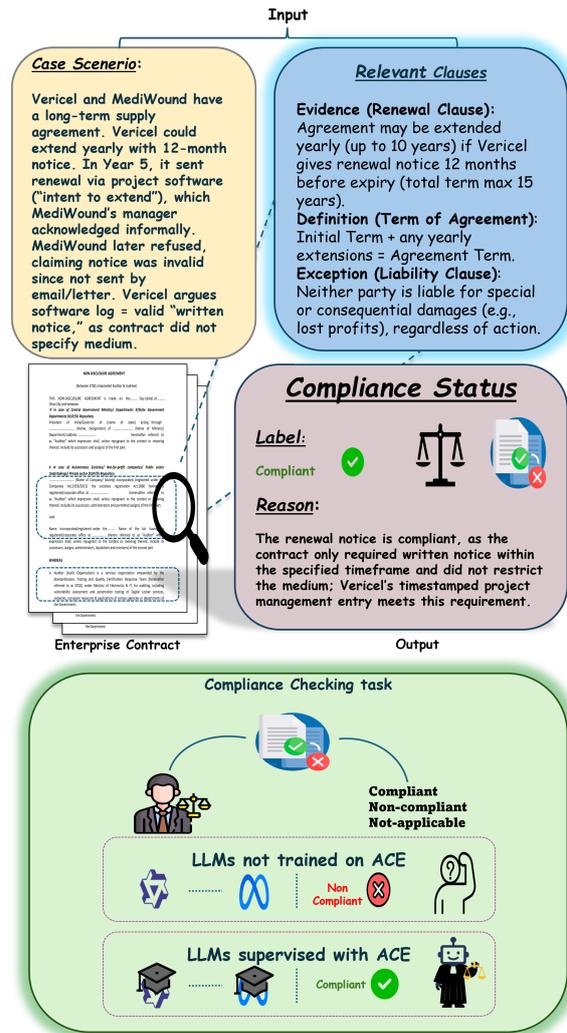


Figure 1: The task involves evaluating an enterprise case scenario against the corresponding contract to determine whether it is compliant, non-compliant, or not applicable.

and error-prone, particularly as modern contracts often span hundreds of pages and exhibit intricate cross-references and dependencies (Surden, 2018).

Recent advances in legal natural language processing (NLP) have produced specialized datasets for clause-level tasks. ContractNLI (Koreeda and

Manning, 2021) formulates compliance queries as entailment problems over individual clauses, while CUAD (Hendrycks et al., 2021) targets obligation extraction. Other efforts such as LexGLUE (Chalkidis et al., 2021) and LegalBench (Guha et al., 2023) provide multi-task evaluations of legal reasoning. While these resources enable progress on isolated clauses, they fall short of realistic compliance scenarios where obligations, conditions, and exceptions interact across multiple clauses. For example, a renewal clause may extend obligations defined elsewhere, or a liability cap may override indemnification duties specified in different sections. Current benchmarks do not test whether models can capture such cross-clause dependencies, limiting their utility for practical compliance applications as shown in figure 1.

To address this gap, we introduce COMPACT (COMpliance PARalegals via Clause graph reasoning over conTRACTs), a framework for training specialized language models that can perform multi-clause compliance reasoning. Our approach rests on two key insights: first, that cross-clause dependencies can be systematically captured through structured clause graphs; and second, that these graphs enable the generation of realistic compliance scenarios that current benchmarks lack.

We develop a pipeline that extracts legal elements from contract clauses and organizes them into clause graphs capturing cross-clause dependencies. From these graphs, we generate the ACE (Assessing Compliance in Enterprise) benchmark: compliance scenarios requiring multi-clause reasoning, including conflict detection, exception handling, and counterfactual analysis.

Given the complexity of these scenarios and enterprise demand for efficient specialized models, we train compact language models on ACE through distillation from larger reasoning models. Our experiments reveal substantial performance gaps between single-clause and multi-clause reasoning tasks, while demonstrating that training on ACE improves compliance capabilities without sacrificing general legal reasoning performance.

Our contributions are:

- We introduce a clause graph framework for modeling cross-clause dependencies and automatically generating compliance scenarios that require multi-hop reasoning.
- We present ACE (Assessing Compliance in Enterprise), the first benchmark of contract-

based compliance scenarios requiring multi-clause reasoning, comprising 4700 scenarios from 633 contracts.

- Through evaluation of state-of-the-art LLMs, we demonstrate that multi-clause reasoning represents a fundamental challenge (34-57% base model accuracy), while training on ACE yields substantial improvements on compliance tasks (+22-43 points) and improves performance on other general legal reasoning benchmarks (PrivaCI-Bench, ContractNLI).

2 Related Work

Theoretical Foundations: Deontic Logic in Legal Compliance. Our approach builds on established frameworks in legal compliance and deontic logic. Governatori et al. (2007) classify legal obligations based on temporal semantics: *achievement obligations* require conditions to occur at least once before a deadline, *maintenance obligations* require conditions to hold continuously until a deadline, and *punctual obligations* require actions at specific times. These temporal distinctions are critical for compliance verification, as they determine how violations are detected and when obligations are satisfied. Building on this classification, Giordano et al. (2013) formalize temporal obligation patterns using deontic logic combined with Answer Set Programming, demonstrating computational verification across diverse domains including HIPAA compliance. Their framework identifies deadline compliance checking, violation time-point identification, and obligation sequencing as essential capabilities for automated compliance systems. These theoretical foundations directly motivate our entity extraction methodology, which decomposes clauses into deontic-temporal components (Subject, Deontic, Action, Object, Temporal_Condition) to capture the obligation structures formalized in this literature (see Appendix B.1 for our complete framework).

Graph-Based Legal Reasoning. Recent work has explored structured representations for legal reasoning through graph-based approaches. Dasgupta et al. (2025) presents OLG++ which constructs rule-level graphs for individual legal provisions using deontic logic primitives, modeling fine-grained obligation structures within single rules. Their approach operates at a complementary granularity to ours: while OLG++ models internal obligation structure within provisions, our clause graphs

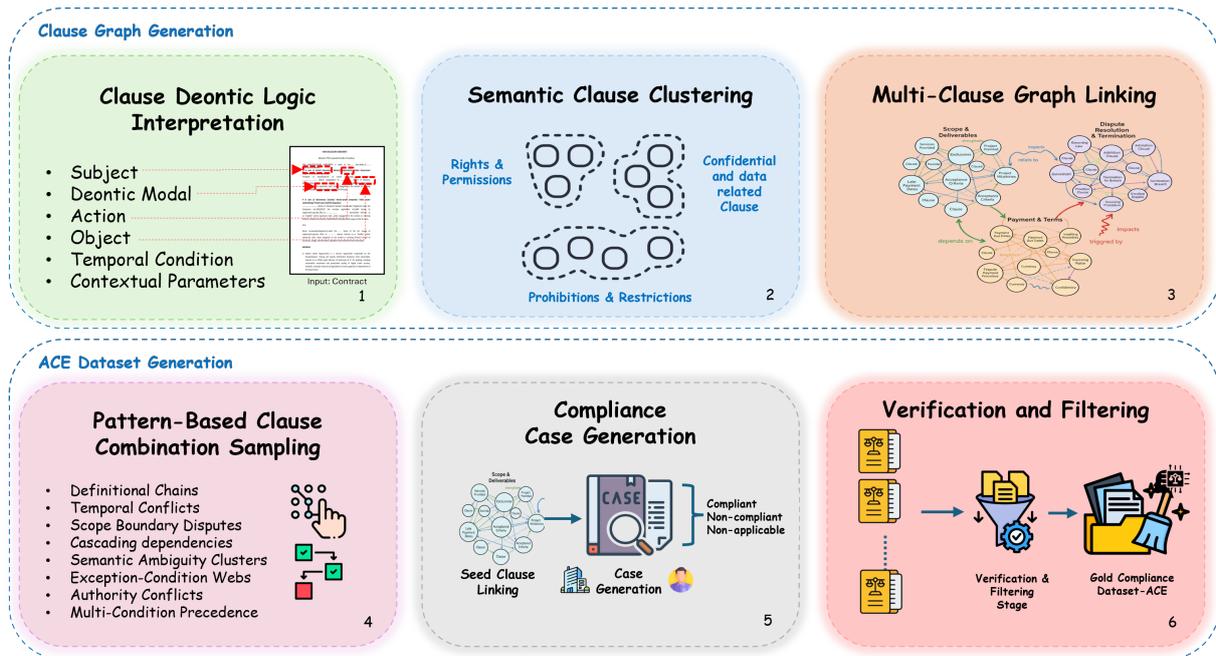


Figure 2: Dataset Generation pipeline for Multi-clause Legal dataset ACE

capture semantic and logical relationships between clauses across entire documents. This document-level perspective enables reasoning about definitional dependencies (where terms defined in one clause govern obligations in another), exception hierarchies (where carve-out clauses modify general rules), and temporal sequences (where obligations cascade across clause boundaries). Our graph construction methodology (Appendix B.3) identifies six core relationship types—DEFINES, COMPLEMENTS, EXCEPTIONS, DEPENDS_ON, CONFLICTS, and SEQUENCE—that generalize across diverse contractual domains.

Domain-Specific Feature Frameworks.

Domain-specific feature frameworks have shown promise for grounding LLMs in specialized legal tasks. GOLDCOIN (Fan et al., 2024) achieves 97.66% accuracy on HIPAA privacy compliance by leveraging Contextual Integrity Theory (Nissenbaum, 2004), which models information flows through five entities including sender, recipient, and information type. However, these information-flow primitives are structurally limited to scenarios where information changes hands, and cannot represent the broader landscape of contractual obligations such as performance requirements, payment terms, or termination rights. Our deontic-temporal entity framework addresses this limitation by abstracting obligations into their core components (Subject, Deontic, Action,

Object, Temporal_Condition), enabling representation of both privacy-specific information flows and general contractual actions. Our framework is applied across 633 contracts from CUAD (Hendrycks et al., 2021) and ContractNLI (Koreeda and Manning, 2021), spanning diverse contractual domains (see Appendix B.1.1).

Existing synthetic data generation methods for legal AI, including GOLDCOIN’s privacy scenario generation, exhibit systematic limitations. First, they are typically domain-specific and focus on single-clause reasoning rather than multi-clause interactions. Second, they suffer from distribution shift: prior work on contextual integrity-based generation shows a 12% performance gap between synthetic and real evaluation cases (Li et al., 2025a), with models achieving 92.64% on synthetic scenarios but only 80.29% on real court cases. This gap stems from detectable artifacts such as explicit feature enumeration and compliance-indicative tonality. Our adversarial scenario generation strategy (see Appendix C.2) addresses both limitations by creating multi-clause compliance scenarios that are technically compliant or non-compliant while appearing otherwise to surface-level analysis, forcing models to perform multi-hop legal reasoning rather than pattern matching.

Comparison with Prior Frameworks Our deontic-temporal entity framework differs from prior legal knowledge graph structures in three key

ways:

(1) Deontic and temporal concepts as explicit entities: We model obligation/permission/prohibition and temporal qualifiers (deadlines, durations, triggers) as structured first-class entities for every clause, rather than implicit attributes or free text.

(2) Document-level clause relationships: While OLG++ (Dasgupta et al., 2025) focuses on rule-level graphs for individual legal provisions, our framework operates at the document level, using edges such as DEFINES, COMPLEMENTS, EXCEPTIONS, DEPENDS_ON, CONFLICTS, and SEQUENCE. This enables reasoning across clauses (e.g., exception stacks, cascading definitions, multi-clause temporal conditions), rather than only within isolated provisions.

(3) Broader applicability beyond domain-specific models: GOLDCOIN’s (Fan et al., 2024) Contextual Integrity schema (Sender–Recipient–Information Type) is tailored to privacy scenarios and cannot represent broader contractual constructs such as obligations, payment terms, or IP licensing. In contrast, our Subject–Deontic–Action–Object abstraction generalizes across a wide range of clause types and contract categories.

3 COMPACT Framework

Our framework constructs clause graphs from contracts to generate compliance scenarios requiring multi-clause reasoning. Figure 2 illustrates the complete pipeline. Our approach consists of two main stages:

1. **Clause Graph Construction** extracts deontic-temporal entities from individual clauses, clusters semantically related clauses, and infers typed relationships between clauses to build structured graph representations (Section 3.1)
2. **ACE Dataset Generation** samples structurally complex clause combinations from these graphs, generates adversarial compliance scenarios, and validates them via LLM-based quality control (Section 3.2).

3.1 Clause Graph Construction

Legal compliance verification requires reasoning about cross-clause dependencies that single-clause benchmarks cannot capture (Governatori et al., 2007; Giordano et al., 2013). A clause defining “Confidential Information” governs obligations elsewhere requiring its protection; an exception clause

modifies general rules stated in other provisions; temporal conditions in one clause trigger obligations specified in another. We address this through structured clause graphs that explicitly model these inter-clause relationships.

3.1.1 Deontic-Temporal Entity Framework

Our entity extraction process uses carefully designed prompts with GPT-4o to systematically decompose each legal clause into structured entities. Building on temporal obligation formalization in legal compliance literature (Governatori et al., 2007; Giordano et al., 2013), we decompose each legal clause into structured entities: Subject (the legal actor bound by the obligation), Deontic (obligation strength: mandatory, permitted, prohibited), Action (the regulated behavior), Object (what the action applies to), and Temporal_Condition (time-based constraints such as deadlines or durations).

This deontic-temporal abstraction enables representation of diverse contractual obligations including performance requirements, payment terms, termination rights, and privacy-related information flows. The framework is validated across 633 contracts from CUAD (Hendrycks et al., 2021) spanning 41 clause types and ContractNLI (Koreeda and Manning, 2021). Details on entity extraction methodology, prompts, and examples are provided in Appendix B.1.

Entity Extraction Validation To ensure reliability, we assessed 50 randomly sampled clauses using both Claude Sonnet 4 and a human reviewer. Table 1 shows the validation results.

Evaluation Type	Precision	Recall
LLM evaluation	97.4%	96.7%
Human evaluation	95.8%	93.8%

Table 1: Entity extraction validation results on 50 randomly sampled clauses.

3.1.2 Semantic Clustering

Entity extraction produces independently annotated clauses. To enable efficient graph construction, we group semantically related clauses into functional clusters using an entity-grounded approach. The clustering leverages extracted entity patterns (Subject, Action, Object, clause_type, contains_defined_terms) combined with legal domain knowledge to identify clauses serving sim-

ilar regulatory functions (e.g., information protection obligations, definitional provisions, exception clauses). Unlike strict partition-based methods, our approach supports multi-cluster membership, acknowledging that individual clauses often serve multiple legal functions simultaneously (Lu et al., 2011). This semantic organization facilitates downstream edge inference by grouping clauses that are likely to exhibit intra-cluster relationships (complementary obligations, temporal sequences) while preserving the structure needed to identify inter-cluster dependencies (definitional links, exception hierarchies). Clustering methodology, prompt design, and output analysis are detailed in Appendix B.2.

3.1.3 Graph Formalization

Following entity extraction and semantic clustering of related clauses, we construct clause graphs through typed edge inference. Formally, let $G = (V, E, R)$ be a clause graph where:

- $V = \{c_1, \dots, c_n\}$ is the set of clause nodes, each annotated with extracted entities
- $E \subseteq V \times V$ is the set of directed edges between clauses
- $R : E \rightarrow \mathcal{T}$ maps edges to relationship types

where

$$\mathcal{T} = \{\text{DEFINES, COMPLEMENTS, EXCEPTIONS, DEPENDS_ON, CONFLICTS, SEQUENCE}\}$$

Edge inference operates on entity-level attributes: **DEFINES** edges connect definitional clauses to operative clauses containing those terms (matching `contains_defined_terms`); **DEPENDS_ON** edges capture temporal or conditional prerequisites (matching `Temporal_Condition` and `Contextual_Parameters`); **EXCEPTIONS** edges link carve-out clauses to the general rules they modify. This document-level perspective complements recent work on rule-level graphs (Dasgupta et al., 2025), enabling reasoning about definitional dependencies, exception hierarchies, and temporal sequences that span clause boundaries. Complete relationship taxonomy and linking methodology are detailed in Appendix B.3.

Clause Graph Validation We evaluated 10 randomly sampled contract-level graphs on two metrics using both Claude Sonnet 4 and a human reviewer. *Cluster Assignment Accuracy* measures how accurately the model groups clauses into their

correct semantic clusters, while *Edge Correctness Accuracy* evaluates whether the graph correctly identifies the relations (edges) between pairs of clauses.

Evaluation Type	Cluster Assignment	Edge Correctness
LLM accuracy	99.1%	93.7%
Human accuracy	96.1%	91.4%

Table 2: Clause graph validation results on 10 randomly sampled contract-level graphs.

3.2 ACE Dataset Generation

Not all clause combinations pose equal interpretive difficulty. We employ pattern-based sampling to identify structurally complex clause combinations, generate adversarial scenarios exhibiting controlled ambiguity, and validate scenario quality via LLM-based filtering.

3.2.1 Pattern-Based Complexity Sampling

We identify eight complexity patterns where legal ambiguity arises, including *definitional chains* (cascading term dependencies), *temporal conflicts* (overlapping deadline obligations), *scope boundary disputes* (unclear permission/restriction boundaries), and *exception-condition webs* (multiple interacting carve-outs). For each clause combination, we compute an enhanced complexity score:

$$\text{Score}_{\text{enh}} = (\text{Score}_{\text{base}} + w_e H + w_c D_{\text{cross}} + w_d D_{\text{net}}) \cdot \phi_{\text{conf}} \quad (1)$$

where $\text{Score}_{\text{base}}$ is the rule-based pattern score, H is relationship type entropy ($H = -\sum_r p_r \log_2 p_r$), D_{cross} is cross-cluster edge density, D_{net} is network density (Newman, 2010), and ϕ_{conf} is average extraction confidence. This formulation combines symbolic pattern detection with graph-theoretic structural metrics to quantify interpretive difficulty. Pattern detection algorithms and complexity analysis are provided in Appendix C.1.

3.2.2 Adversarial Scenario Generation

As discussed in Section 2, synthetic data generated using prior approaches when used for training models exhibits systematic performance gaps due to detectable distributional artifacts (Li et al., 2025a). Our generation strategy leverages adversarial prompting to construct scenarios that closely resemble genuine compliance cases, yet exhibit

subtle inconsistencies that challenge surface-level interpretation. We generate three scenario types:

Compliant-with-Distractor scenarios where actions strictly satisfy requirements but are masked by misleading contextual details (e.g., informal communication channels, minimalist compliance)

Violation-with-Plausible-Defense scenarios containing clear breaches masked by superficially reasonable justifications (e.g., good-faith intent, scope misapplication)

Non-Applicable via Cross-Matching scenarios that pair clause combinations with scenarios originally generated for non-overlapping provisions from the same document, ensuring domain relevance while avoiding trivial negatives. This approach forces evaluation systems to perform multi-hop legal reasoning rather than pattern matching on synthetic artifacts. Generation strategies and example scenarios are detailed in Appendix C.2.

3.2.3 LLM-Based Validation

Generated scenarios undergo validation via Claude Sonnet 4, distinct from our generation model (GPT-4o), to mitigate self-preference bias (Panickssery et al., 2024; Verga et al., 2024). Following reference-guided evaluation approaches that reduce self-enhancement bias (Badshah and Sajjad, 2024), the validator receives the scenario, governing clauses, claimed label, and provided reasoning. Validation assesses mainly two dimensions: (1) **Clause Governance** (relevance score ≥ 0.7): whether the scenario’s central legal conflict is meaningfully governed by the provided clauses; (2) **Label Correctness** (confidence score ≥ 0.8): whether the compliance label accurately reflects clause application to scenario facts. Scenarios must satisfy both checks for acceptance. Complete validation protocol is provided in Appendix C.3.

3.3 Model Training via Distillation

We employ a **distillation-based training pipeline** to transfer structured legal reasoning abilities from a powerful reasoning model to smaller, instruction-tuned LLMs. Specifically, we use the **DeepSeek-R1-0528** (Guo et al., 2025) model as the *teacher* to generate detailed reasoning traces for each scenario in the **ACE (Assessing Compliance in Enterprise)** dataset. Each trace provides a step-by-step explanation of how the given *Regulatory Provisions* relate to the *Case Scenario*, concluding with a compliance judgment.

To ensure the correctness of the supervision data,

we perform **rejection sampling**—filtering out instances where the reasoning or final label contradicts the clause evidence. Only validated reasoning-label pairs are retained for model training. (see Appendix C.4)

We fine-tune Llama-3.2 (3B), Llama-3.1 (8B) (Touvron et al., 2023), and Qwen2.5 (1.5B, 3B, 7B) instruction models (Bai et al., 2024)¹ using **Supervised Fine-Tuning (SFT)** with **LoRA (Low-Rank Adaptation)** for parameter-efficient training, as implemented in the TRL framework.

All models were trained with **bfloat16 precision** and **gradient checkpointing** on one NVIDIA H100 GPUs, with each model taking approximately 1-2 hours to train. Detailed hyperparameter settings are provided in the Appendix C.4.1.

4 ACE (Assessing Compliance in Enterprise) Dataset

4.1 Corpus Statistics

The ACE dataset comprises 633 legal documents spanning 26 distinct agreement types, sourced from CUAD and ContractNLI repositories. As illustrated in Figure 3, the distribution reflects realistic enterprise contract frequencies. NDAs constitute the largest category at 19.4% (123 documents), followed by Maintenance Agreements (5.4%), License Agreements (5.2%), and Distributor Agreements (5.1%). The dataset also includes lower-frequency but practically relevant categories such as Consulting Agreements (1.7%) and Non-Disparagement Agreements (0.5%), enabling evaluation across both high-resource and long-tail compliance scenarios.

4.2 Label Distribution

The dataset contains 4,700 clause-scenario pairs distributed across training (3,240), validation (360), and test (1,100) splits, as detailed in Table 3. The three compliance categories are approximately balanced: Compliant (1,580 instances, 33.6%), Non-Compliant (1,600 instances, 34.0%), and Non-Applicable (1,520 instances, 32.3%). This balanced design mitigates class imbalance issues while reflecting the practical occurrence of non-applicable clauses in real-world contract analysis.

4.3 Dataset Diversity

To assess the diversity of generated scenarios in our dataset, we employ a combined diversity met-

¹All models are available at <https://huggingface.co/>

Comprehensive Legal Document Distribution Analysis

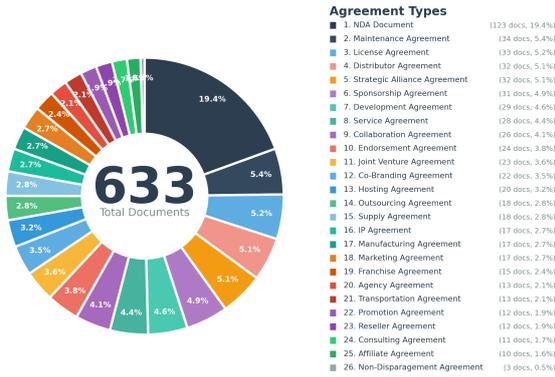


Figure 3: Distribution of legal document types in the ACE Dataset (N=633 documents across 26 agreement categories) from CUAD and ContractNLI.

Labels	Train	Val	Test	Total
Compliant	1080	120	380	1580
Non-Compliant	1080	120	400	1600
Non Applicable	1080	120	320	1520
Total	3240	360	1100	4700

Table 3: Distribution of output labels across train and test split on ACE dataset

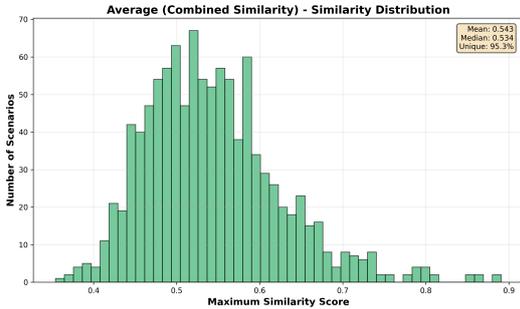


Figure 4: Distribution of pairwise similarity scores across all scenarios using combined ROUGE-L and embedding-based metrics.

ric that captures both lexical and semantic variations. Following established practices in text diversity measurement (Lin, 2004; Shaib et al., 2024), we calculate lexical similarity using ROUGE-L and semantic similarity using sentence embeddings via the all-mpnet-base-v2 model (Reimers and Gurevych, 2019) and took average of them. Recent work has demonstrated that combining multiple diversity metrics provides complementary perspectives, as lexical measures capture surface-level variation while embedding-based approaches capture semantic distinctiveness (Shaib et al., 2024; Zhu et al., 2018). Using a similarity threshold

of 0.7, our combined diversity metric achieves a 95.3% uniqueness rate, indicating that the vast majority of scenario pairs are distinct while maintaining expected thematic overlap within the legal domain. The complete distribution of similarity scores across all scenario pairs in test set is presented in Figure 4.

4.4 Dataset Quality Assessment

To evaluate the quality and realism of our synthetically generated scenarios, we adopt a discriminator-based approach inspired by adversarial training frameworks (Goodfellow et al., 2014), where we employ an LLM to distinguish between real and synthetic samples. Following recent studies showing that Claude models align closely with human judgments on evaluation tasks (Verga et al., 2024), we leveraged Claude Sonnet 4 to evaluate our dataset along three key dimensions:

- Synthetic Detectability:** This evaluates whether the model can identify scenarios as synthetic based on linguistic or structural features, serving as an indicator of their naturalness and realism.
- Compliance Interpretability:** Here, the model classifies scenarios as *compliant*, *violation*, or *unclear*, using only the narrative text *without* access to the governing regulatory clauses. Low performance on this task supports our hypothesis that accurate compliance assessment requires explicit regulatory context, rather than being solvable through superficial patterns.
- Paired Comparison:** In this evaluation, the model distinguishes synthetic scenarios from real legal cases: HIPAA (Li et al., 2025b) in direct head-to-head comparisons, providing a robust measure of realism relative to authentic legal data.

Table 4 reports evaluation results; lower accuracy indicates better scenario quality.

Dataset	Synthetic Detection	Compliance	Paired Comparison
ACE	18.8%	38.6%	57.1%

Table 4: Quality assessment results of the ACE dataset. Note that lower accuracy corresponds to higher scenario quality and realism. Compliance assessment uses only the narrative text without access to regulatory clauses

Human Expert Evaluation In addition to model-based evaluation, we conducted comprehensive legal human expert assessment to validate ACE

dataset quality. We evaluated 50 randomly sampled ACE scenarios with a legal expert.

Evaluation Setup: For each datapoint, the expert was given the generated compliance case scenario and the specific regulatory clauses from which it was constructed.

Evaluation Dimensions: The expert evaluated each scenario along two dimensions:

- **Clause governance** – whether the cited regulatory clause actually governs the given legal case scenario.
- **Label correctness** – whether the assigned label (Compliant / Non-Compliant / Not-Applicable) is correct or not.

Evaluation Dimension	Correct Cases	Accuracy
Clause governance	48 / 50	96%
Label correctness	46 / 50	92%

Table 5: Human expert evaluation results on 50 randomly sampled ACE scenarios.

Cases with incorrect labels were primarily those affected by ambiguous clause wording and conflicts between overlapping regulatory requirements. The expert confirmed that the reviewed scenarios appear realistic and plausible, validating the quality of our adversarial generation approach. Overall, this evaluation confirms that the dataset exhibits strong consistency, relevance, and legal validity.

5 Experimental Setup

We evaluate our fine-tuned models across three dimensions to assess both in-domain performance and out-of-distribution generalization. First, we establish baseline performance on the ACE test set, where models are trained and tested on contract compliance checking. Second, to evaluate **cross-domain generalization**, we test our models on PrivaCI-Bench’s (Li et al., 2025b) EU AI Act and HIPAA subsets, representing AI regulation and healthcare compliance respectively—domains distinct from the contract law on which our models were trained. Following our training distribution (average token length $\sim 6k$), we filter samples exceeding 6k tokens, yielding 2,617 EU AI Act samples (out of 3,000) and 120 HIPAA samples (out of 214). Third, to evaluate **cross-task generalization**, we test on ContractNLI (Koreeda and Manning, 2021), which requires legal entailment reasoning rather than compliance checking. Unlike prior work (e.g., GOLDCOIN (Fan et al.,

2024), Domain01s (Chu et al., 2025)) that evaluates primarily within the training domain, our experimental design explicitly tests whether models learn generalizable legal reasoning patterns that transfer across both legislative domains and task types.

Evaluation Metrics: Across all datasets, we report Accuracy and macro-F1 to balance overall correctness with sensitivity to minority classes.

6 Result and Analysis

In-domain Performance on ACE. Table 6 reports results on the ACE test set, assessing models trained and evaluated within the contract compliance domain. Fine-tuning on ACE consistently improves both Accuracy and macro-F1 across all models. Notably, even small language models such as LLaMA-3B and Qwen-1.5B achieve competitive performance gain of +51 and +38.2 % points respectively in F1-score after fine-tuning, indicating that ACE provides sufficient structure for learning cross-clause dependencies without requiring large-scale capacity.

Cross-domain Generalization. We next evaluate transfer to the EU AI Act and HIPAA subsets of PrivaCI-Bench (Li et al., 2025b), which represent distinct regulatory regimes in AI governance and healthcare (Table 7). Models fine-tuned on ACE shows clear gains over their base versions, indicating that clause-level reasoning learned from contracts transfers effectively to diverse regulatory domains. The Qwen models, in particular, demonstrate stable cross-domain generalization, while slight drops in larger models suggest the need for better calibration across domains.

Cross-task Generalization. To examine whether ACE fine-tuning benefits broader legal reasoning beyond compliance classification, we evaluate on ContractNLI (Koreeda and Manning, 2021), a legal entailment task requiring inference over interdependent clauses (Table 8). Fine-tuned models consistently outperform their base counterparts in F1-score, indicating that the relational reasoning learned in ACE transfers effectively to entailment tasks, bridging compliance verification and legal inference.

Overall Insights. Across all evaluation dimensions, ACE fine-tuning delivers substantial in-domain gains, consistent cross-domain transfer, and measurable improvements in cross-task reasoning. These results demonstrate that models fine-tuned with ACE learns transferable legal reasoning

Model	Accuracy			F1-Score		
	Base	FT	Δ	Base	FT	Δ
LLaMA-3B	42.4	78.9	+36.5	28.3	79.3	+51.0
LLaMA-8B	34.5	77.6	+43.1	17.5	78.5	+61.0
Qwen-1.5B	45.2	73.9	+28.7	36.9	75.1	+38.2
Qwen-3B	42.5	75.8	+33.3	28.7	77.5	+48.8
Qwen-7B	56.8	78.5	+21.7	53.1	78.9	+25.8

Table 6: Performance comparison of base and fine-tuned models on the ACE test set.

Model	Type	EU AI Act		HIPAA	
		Acc	F1	Acc	F1
LLaMA-3B	Base	37.2	20.1	37.8	30.9
	FT	44.6 \uparrow	40.2 \uparrow	44.5 \uparrow	34.8 \uparrow
LLaMA-8B	Base	62.1	54.3	42.1	40.4
	FT	59.5 \downarrow	55.2 \uparrow	34.8 \downarrow	33.7 \downarrow
Qwen-1.5B	Base	45.3	33.2	27.3	23.3
	FT	50.8 \uparrow	49.3 \uparrow	53.7 \uparrow	40.4 \uparrow
Qwen-3B	Base	66.2	53.6	46.5	38.1
	FT	66.8 \uparrow	66.2 \uparrow	57.3 \uparrow	44.7 \uparrow
Qwen-7B	Base	74.1	64.9	71.5	55.3
	FT	76.8 \uparrow	75.7 \uparrow	72.8 \uparrow	60.8 \uparrow

Table 7: Performance comparison across PrivaCI-Bench’s EU AI Act and HIPAA task subsets.

patterns—capturing obligations, exceptions, and conditional dependencies that generalize beyond contract texts. The strong performance of small language models (e.g., LLaMA-3B, Qwen-1.5B) shows ACE’s practicality for enterprise use, enabling reliable compliance reasoning with minimal computational overhead.

Model	Accuracy		F1-Score	
	Base	FT	Base	FT
LLaMA-3B	12.8	25.1 \uparrow	10.6	25.4 \uparrow
LLaMA-8B	45.4	41.8 \downarrow	29.1	41.2 \uparrow
Qwen-1.5B	18.1	22.5 \uparrow	17.8	22.7 \uparrow
Qwen-3B	53.1	56.9 \uparrow	47.8	52.1 \uparrow
Qwen-7B	62.5	64.2 \uparrow	59.6	58.6 \downarrow

Table 8: Performance comparison of base and fine-tuned models on the ContractNLI test set.

7 Summary and Conclusion

Real-world contract compliance requires reasoning across multiple clauses, a capability neglected by existing legal AI benchmarks. We address this with the COMPACT framework, which models contracts as clause graphs to capture cross-clause de-

pendencies. From these graphs, we construct the ACE benchmark. Our key finding is that while multi-clause reasoning poses a significant challenge to state-of-the-art models (34-57% accuracy), training on ACE yields substantial gains (+22-43 % points), demonstrating that our approach successfully captures transferable legal reasoning skills. This work establishes the first benchmark for systematic multi-clause contract compliance.

Limitations and Future Work

While COMPACT and ACE advance multi-clause compliance reasoning, several limitations remain. Our dataset focuses on English-language enterprise contracts, limiting coverage across jurisdictions and languages.

The models are trained on sequences averaging 6,000 tokens, restricting evaluation to (Li et al., 2025b) tasks within this context length. Moreover, our training could benefit from systematic hyperparameter tuning and reinforcement learning to enhance generalization. The reasoning traces derived from DeepSeek’s think tokens can also be improved through task-specific reasoning template tailored for legal compliance analysis.

Future work should explore multilingual and longer-context settings, along with reinforcement learning-based optimization to enhance reasoning robustness and cross-domain generalization. Our framework assumes that relevant clauses are pre-identified; retrieving these spans from contracts remains an important open challenge for future work.

References

1996. [Health Insurance Portability and Accountability Act of 1996](#). Pub. L. 104–191, 110 Stat. 1936. Federal law mandating national standards for the security and privacy of health data.
- Sher Badshah and Hassan Sajjad. 2024. Reference-guided verdict: Llms-as-judges in automatic evaluation of free-form text. *arXiv preprint arXiv:2408.09235*.
- Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, and 1 others. 2024. Qwen technical report. *arXiv preprint arXiv:2309.16609*.
- Ilias Chalkidis, Abhik Jana, Dirk Hartung, Michael Bommarito, Ion Androutsopoulos, Daniel Martin Katz, and Nikolaos Aletras. 2021. Lexglue: A benchmark dataset for legal language understanding in english. *arXiv preprint arXiv:2110.00976*.
- Xu Chu, Zhijie Tan, Hanlin Xue, Guanyu Wang, Tong Mo, and Weiping Li. 2025. Domaino1s: Guiding llm reasoning for explainable answers in high-stakes domains. *arXiv preprint arXiv:2501.14431*.
- Subhasis Dasgupta, Jon Stephens, and Amarnath Gupta. 2025. Olg++: A semantic extension of obligation logic graph. *arXiv preprint arXiv:2507.05488*.
- Wei Fan, Haoran Li, Zheyang Deng, Weiqi Wang, and Yangqiu Song. 2024. [Goldcoin: Grounding large language models in privacy laws via contextual integrity theory](#). *Preprint*, arXiv:2406.11149.
- Laura Giordano, Alberto Martelli, and Daniele Theseider Dupré. 2013. [Temporal deontic action logic for the verification of compliance to norms in asp](#). In *ICAIL*, pages 53–62.
- Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial nets. In *Advances in neural information processing systems*, volume 27, pages 2672–2680.
- Guido Governatori. 2010. [Law, logic and business processes](#). In *2010 Third International Workshop on Requirements Engineering and Law*, pages 1–10.
- Guido Governatori, Joris Hulstijn, Régis Riveret, and Antonino Rotolo. 2007. Characterising deadlines in temporal modal defeasible logic. In *AI 2007: Advances in Artificial Intelligence*, pages 486–496, Berlin, Heidelberg. Springer Berlin Heidelberg.
- 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, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, 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*.
- Yuta Koreeda and Christopher D Manning. 2021. Contractnli: A dataset for document-level natural language inference for contracts. *arXiv preprint arXiv:2110.01799*.
- Haoran Li, Wei Fan, Yulin Chen, Cheng Jiayang, Tianshu Chu, Xuebing Zhou, Peizhao Hu, and Yangqiu Song. 2025a. [Privacy checklist: Privacy violation detection grounding on contextual integrity theory](#). In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 1748–1766, Albuquerque, New Mexico. Association for Computational Linguistics.
- Haoran Li, Wenbin Hu, Huihao Jing, Yulin Chen, Qi Hu, Sirui Han, Tianshu Chu, Peizhao Hu, and Yangqiu Song. 2025b. Privaci-bench: Evaluating privacy with contextual integrity and legal compliance. *arXiv preprint arXiv:2502.17041*.
- Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.
- Qiang Lu, Jack G. Conrad, Khalid Al-Kofahi, and William Keenan. 2011. [Legal document clustering with built-in topic segmentation](#). In *Proceedings of the 20th ACM International Conference on Information and Knowledge Management, CIKM '11*, pages 383–392. ACM.
- Mark Newman. 2010. *Networks: an introduction*. Oxford University Press.
- Helen Nissenbaum. 2004. Privacy as contextual integrity. *Washington Law Review*, 79(1):119–158.
- Arjun Panickssery, Samuel R. Bowman, and Shi Feng. 2024. Llm evaluators recognize and favor their own generations. *arXiv preprint arXiv:2404.13076*.
- 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. Association for Computational Linguistics.

- Chantal Shaib, Joe Barrow, Jiuding Sun, Alexa F. Siu, Byron C. Wallace, and Ani Nenkova. 2024. Standardizing the measurement of text diversity: A tool and a comparative analysis of scores. *arXiv preprint arXiv:2403.00553*.
- Harry Surden. 2018. Artificial intelligence and law: An overview. *Ga. St. UL Rev.*, 35:1305.
- Richard Susskind and Daniel Susskind. 2017. The future of the professions: How technology will transform the work of human experts. *Journal of Nursing Regulation*, 8(2):52.
- Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, and 1 others. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.
- Pat Verga, Sebastian Hofstätter, Sophia Althammer, Yixuan Su, Aleksandra Piktus, Arkady Arkhangorodsky, Minjie Xu, Naomi White, and Patrick Lewis. 2024. Replacing judges with juries: Evaluating llm generations with a panel of diverse models. *arXiv preprint arXiv:2404.18796*.
- Yaoming Zhu, Sidi Lu, Lei Zheng, Jiaxian Guo, Weinan Zhang, Jun Wang, and Yong Yu. 2018. Texygen: A benchmarking platform for text generation models. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*.

A Appendix

In this appendix, we provide comprehensive technical details and methodological specifications that complement the main paper. Due to space constraints in the main text, we present here the complete implementation of our three-stage pipeline for legal document analysis and compliance evaluation.

In particular, this appendix contains the following:

- **Graph of Clause Generation**
 - Entity Extraction for Legal Clause Analysis
 - * Design Rationale: From Privacy to General Compliance
 - * Deontic-Temporal Entity Framework
 - * Clause Type Classification for Graph Construction
 - * Entity Extraction Prompt Design
 - Semantic Clustering of Legal Clauses
 - * Motivation: From Flat Entities to Relational Structure
 - * Clustering Methodology
 - * Multi-Cluster Membership Framework
 - * Clustering Prompt Architecture
 - * Clustering Output Analysis
 - Graph Construction via Clause Linking
 - * Motivation: From Clusters to Graphs
 - * Relationship Taxonomy and Two-Phase Approach
 - * Linking Methodology
 - * Prompt Architecture
 - * Output Examples and Visualization
- **Case Generation for Compliance Evaluation**
 - Pattern-Based Clause Combination Sampling
 - Adversarial Scenario Generation
 - * Design Rationale: Avoiding Synthetic Distribution Shift
 - * Scenario Generation Strategy
 - * Generation Prompts
 - LLM-Based Validation
- **Model Training Details**
 - Training Configuration
- **Compliance Evaluation Task**
- **Case Study on Model Performance**

B Graph of Clause Generation

Our graph generation approach employs a systematic three-stage pipeline designed to transform le-

gal documents into structured clause graphs. The pipeline consists of: (1) **entity extraction** to identify deontic structures and legal components within individual clauses, (2) **semantic clustering** to group related clauses based on extracted features, and (3) **graph construction via clause linking** to detect relationships both within clusters and across different thematic groups. Each stage builds upon the structured outputs of the previous stage, enabling scalable and interpretable legal document analysis.

B.1 Entity Extraction for Legal Clause Analysis

B.1.1 Design Rationale: From Privacy to General Compliance

Recent approaches to legal AI have demonstrated that domain-specific feature frameworks can effectively ground LLMs in specialized legal tasks. GOLDCOIN (Fan et al., 2024) achieves 97.66% accuracy on HIPAA (Act, 1996) privacy compliance by leveraging Contextual Integrity Theory (Nissenbaum, 2004), which models information flows through five core entities: *Sender*, *Recipient*, *Subject*, *Information Type*, and *Transmission Principle*.

However, **privacy obligations represent only a subset of contractual compliance**. Consider this clause from a supply agreement:

“Supplier shall deliver conforming goods within 14 days of purchase order, and Buyer shall inspect goods within 7 days of delivery.”

GOLDCOIN’s information-flow abstraction cannot represent:

- **Non-information obligations:** No information is being transmitted; performance actions are required
- **Temporal dependencies:** Inspection window depends on delivery completion
- **Cascading deadlines:** 14-day + 7-day sequence with implicit start/end triggers
- **Dual actor obligations:** Both parties bound by distinct requirements

Why Deontic-Temporal Entities? Our entity framework addresses three limitations of privacy-centric approaches:

1. Generalizability Across Obligation Types

Contextual integrity features are structurally limited to scenarios where information changes hands.

Our framework is validated on two complementary datasets: 510 contracts from CUAD (Hendrycks et al., 2021) spanning 41 clause types across 25 contract categories, and 123 NDAs from ContractNLI (Koreeda and Manning, 2021).

NDAs as a bridge case: ContractNLI’s 17 hypotheses (Koreeda and Manning, 2021) demonstrate that even privacy-focused contracts require temporal-deontic reasoning beyond information flow. For instance:

- **Temporal obligations:** Hypothesis #13 asks whether “Receiving Party shall destroy or return some Confidential Information upon the termination of Agreement”
- **Persistence of obligations:** Hypothesis #16 examines whether “Some obligations of Agreement may survive termination of Agreement”
- **Conditional permissions:** Hypothesis #7 addresses whether “Receiving Party shall notify Disclosing Party in case Receiving Party is required by law [...] to disclose”

These hypotheses involve *when* obligations apply (temporal logic), *who must do what* (deontic assignment), and *under what conditions* (modal logic)—elements that Contextual Integrity’s information-flow primitives (Sender, Recipient, Subject, Information Type, Transmission Principle) cannot express.

Extending beyond privacy: While NDAs expose limitations even within privacy-adjacent domains, CUAD’s 41 clause types (Hendrycks et al., 2021) encompass entirely non-informational obligations including payment terms (Revenue/Profit Sharing, Minimum Commitment), performance requirements (Post-Termination Services, Audit Rights), intellectual property rights (IP Ownership Assignment, License Grant), liability allocation (Cap on Liability, Liquidated Damages), and termination conditions (Termination for Convenience, Expiration Date).

By decomposing obligations into Subject-Deontic-Action-Object, we subsume privacy information flows (Action $\in \{share, disclose, access\}$) while extending to general contractual actions (Action $\in \{deliver, pay, indemnify, license, terminate\}$).

2. Explicit Temporal Modeling

Legal compliance literature identifies temporal structure as critical for distinguishing obligation types. Governatori et al. (2007) classify obligations based on their temporal semantics:

- **Achievement obligations:** condition must occur *at least once* before deadline
- **Maintenance obligations:** condition must hold *continuously* until deadline
- **Punctual obligations:** action required *at specific time*

These obligation types are recognized as essential for business process compliance verification (Governatori, 2010). Building on this classification, Giordano et al. (2013) provide a formal framework using temporal deontic logic combined with Answer Set Programming, demonstrating how such obligations can be computationally verified across diverse compliance domains including HIPAA.

GOLDCOIN conflates temporal constraints into “Transmission Principles,” losing the granularity needed for compliance verification (Giordano et al., 2013):

- Deadline compliance checking
- Violation time-point identification
- Contrary-to-duty obligation handling
- Obligation sequencing and dependencies

Concrete formalizations from compliance literature:

Giordano et al. (2013) formalize temporal obligation patterns drawn from business process compliance literature (Governatori, 2010):

- **Achievement with deadline:** “Customers must pay before delivery of goods” $\rightarrow O(\neg goods \cup paid)$, where payment must occur at least once before the deadline event
- **Maintenance:** “Keep positive bank balance until charges taken” $\rightarrow O(pos_balance \cup charges_taken)$, requiring the condition to hold *continuously*
- **Punctual:** “System must immediately acknowledge message” $\rightarrow O(\mathbf{X}ack)$, requiring action at the next temporal point
- **Non-preemptive achievement:** “Send contract copy after signing” $\rightarrow O(\neg deadline \cup \langle$

$send_copy \rangle \top$), where the action must occur *after* the triggering event and before the deadline

Such formal temporal distinctions are essential for contractual compliance, where obligations often involve complex deadline dependencies. Our Temporal_Condition field makes deadlines first-class entities, enabling explicit modeling of: “within X days,” “by date Y,” “during term,” “upon event Z.”

3. Compositional Graph Construction

While GOLDCOIN constructs a static hierarchy of legal norms through subsume and refer relationships, this approach treats norms as **isolated terminals** without capturing their interactions. In contractual contexts, clauses frequently exhibit complex interdependencies that GOLDCOIN’s privacy-focused framework cannot represent.

In our model, each clause is a **node**, annotated with its extracted entities (Subject, Deontic, Action, Object, Temporal_Condition). The power of our approach lies in inferring typed **edges** between these nodes by matching their entity attributes. This transforms a flat list of clauses into a relational graph that captures the contract’s logical and temporal structure.

Edge inference is guided by rules that operate on the entity-level:

- **Cross-clause dependency detection:** Modeling how obligations in one clause trigger or modify obligations in others
- **Temporal constraint propagation:** Sequencing obligations based on deadline dependencies and conditional triggers
- **Conflict resolution:** Identifying contradictory requirements across different clause types
- **Obligation chain analysis:** Tracing compliance pathways through interconnected requirements

This compositional approach is not supported by the GOLDCOIN framework. Its entities (*Sender*, *Recipient*, *Information Type*) are too semantically narrow to establish general-purpose links. One cannot infer a temporal dependency or a definition from matching a ‘Sender’ entity alone. Our framework, by abstracting obligations to their core components, enables the detection of these critical

inter-clause relationships, which is essential not only for holistic compliance analysis but also for other legal tasks such as contract interpretation, understanding, and relevant clause retrieval.

Empirical Validation of Generalizability

Cross-domain coverage comparison:

- **GOLDCOIN (Contextual Integrity Theory):** 1 domain (Healthcare/HIPAA), privacy-related clause types
- **Our Framework:** 25+ contract types (CUAD) + NDA contracts (ContractNLI), 41+ clause types spanning diverse legal domains

The Subsumption Property The generality of our deontic-temporal framework is further highlighted by its ability to subsume domain-specific models like GOLDCOIN’s. A privacy obligation is merely a specific instance of a general contractual obligation where the Action is one of {*share*, *disclose*, *transmit*} and the Object is an Information Type.

Example transformation:

- **GOLDCOIN:** ‘Dr. Smith (Sender) must not disclose (Transmission Principle) patient records (Information Type) to an insurer (Recipient).’
- **Our Framework:** ‘Dr. Smith (Subject) *shall not* (Deontic) *disclose* (Action) *patient records* (Object) to an *insurer* (Counterparty).’

Therefore, our model can represent privacy clauses natively while also capturing the vast landscape of non-privacy obligations that GOLDCOIN cannot. This makes our compositional graph construction a fundamentally more powerful and scalable approach for automated legal document understanding.

B.1.2 Deontic-Temporal Entity Framework

Building on the rationale established in Section B.1.1, we formalize our entity extraction approach using deontic logic (Giordano et al., 2013) as the theoretical foundation.

Core Entity Decomposition Each legal clause imposing or granting rights is decomposed into eight structured fields:

1. **Subject:** The legal actor bound by or benefiting from the obligation

2. **Deontic Modal:** The type and strength of obligation/permission
 - *Obligation:* “shall,” “must,” “agrees to” (mandatory)
 - *Permission:* “may,” “is entitled to” (optional)
 - *Prohibition:* “shall not,” “is prohibited” (forbidden)
 - *Implicit detection:* “No right is granted” → prohibition
3. **Action:** The regulated behavior or activity (e.g., deliver, pay, disclose)
4. **Object:** The thing/information being acted upon (e.g., goods, fees)
5. **Temporal Condition:** Time-based constraints on obligation validity
 - *Deadlines:* “within 30 days,” “by January 1”
 - *Durations:* “for 2 years,” “during the term”
 - *Time periods:* “after termination,” “upon expiration”
6. **Contextual Parameters:** Situational triggers and constraints
 - *Triggering events:* “upon breach,” “if requested”
 - *Purpose:* “for treatment,” “to enforce rights”
 - *Manner:* “in writing,” “via certified mail”
 - *Conditions:* “unless waived,” “except as permitted”
7. **Contains Defined Terms:** Capitalized/quoted terms requiring definition lookup
8. **Counterparty:** The entity receiving/providing the action (if applicable)

This 8-field decomposition provides a *complete* representation of deontic-temporal obligations in contracts. Each field serves a distinct role: Subject, Action, Object, and Counterparty capture the *who-does-what-to-whom* structure; Deontic_Modal encodes obligation strength; Temporal_Condition handles

when constraints; and Contextual_Parameters captures conditional triggers and procedural requirements. The Contains_Defined_Terms flag enables cross-referencing to definitional clauses. Together, these entities formalize the deontic-temporal patterns identified in legal compliance literature (Governatori, 2010; Giordano et al., 2013) while remaining general enough to apply across diverse contractual domains.

B.1.3 Clause Type Classification for Graph Construction

Our pipeline performs automated clause type classification to guide downstream clustering and edge inference. We distinguish four functional categories: **Operative clauses** impose obligations or grant permissions (e.g., "Supplier shall deliver goods") and receive full 8-field entity extraction; **Definitional clauses** establish term meanings (e.g., "Confidential Information means...") and generate *definition edges* to clauses containing those terms; **Exception clauses** create carve-outs from general rules (e.g., "Except when required by law...") and produce *modification edges* to the rules they alter; and **Conditional clauses** specify triggering events (e.g., "Upon termination...") and create *temporal dependency edges* to the consequences they activate. Classification employs pattern-based recognition of linguistic markers ("means," "shall," "except," "upon") combined with contextual disambiguation to distinguish operative prohibitions ("No right is granted") from exception markers ("Obligations will not apply to..."). This typed classification enables the semantic clustering stage (B.2) to group clauses by functional role and the graph linking stage (B.3) to infer relationship-specific edges.

B.1.4 Entity Extraction Prompt Design

The following prompt implements our comprehensive entity extraction methodology for operative legal clauses.

Entity Extraction Prompt

Extract legal entities and deontic structure from the following contract clause:

Clause: {clause_text}

Critical Extraction Guidelines:

Subject Identification: - Identify the legal actor who must

perform/is bound by the obligation
- Distinguish between grammatical subject and legal actor - For prohibitions (e.g., "No right is granted"), the subject is the implicit grantor

Deontic Modal Detection: - Look for explicit modals: "shall", "must", "may", "will", "agree to" - Detect implicit obligations: "No right is granted" = "shall not grant" (prohibition) - For negative constructions, identify the underlying deontic force (e.g., "is prohibited" = "shall not" (obligation))

Object Precision: - Identify what the action directly applies to - Include the complete object of the action - For "grant rights in X" → Object is "rights in X", not just "X"

Counterparty Logic: - If Subject is "The Parties" → counterparty is "reciprocal parties" - If action involves bilateral relationship → identify the other party - Look for explicit mentions: "to the other", "the Discloser", "the Recipient" - For obligations TO someone → that someone is the counterparty

Temporal vs Conditional Distinction:
- **Temporal (time-based):** "within 30 days", "indefinitely", "during term", "for X years" - **Conditional (event-based):** "upon request", "if court ordered", "when terminated", "on demand"

Complex Clause Detection: - Identify clauses with multiple coordinated obligations or permissions - Detect different deontic modals within the same clause - For complex clauses, provide decomposition analysis with reasoning and split constituent clauses

Entity Types to Extract: 1. **Subject:** The legal actor bound by the

obligation (string) 2. **Deontic:** Deontic modal with type (string) 3. **Action:** The primary regulated action (string) 4. **Object:** The complete direct object of the action (string) 5. **Temporal_Condition:** Pure time-based conditions only: deadlines (by/within X), durations (for X period), time periods (during/after X), or 'not specified' (string) 6. **Contextual_Parameters:** Manner, circumstances, purposes, roles, constraints, environments, or triggering events under which the action occurs, or 'not specified' (string) 7. **contains_defined_terms:** Capitalized/quoted terms needing definition (comma-separated string or "none") 8. **counterparty:** Who receives/provides the action (string or "not specified")

Output Schema (JSON)

```
{
  "clause_type": "operative",
  "entities": {
    "Subject": "legal actor bound by obligation",
    "Deontic": "deontic modal with type",
    "Action": "primary regulated action",
    "Object": "complete direct object of action",
    "Temporal_Condition": "time-based conditions",
    "Contextual_Parameters": "event triggers or circumstances",
    "contains_defined_terms": "terms needing definition",
    "counterparty": "action recipient/provider"
  },
  "complexity_analysis": {
    "reasoning": "explanation if complex clause detected",
    "split_clauses": ["clause1", "clause2", "clause3"]
  } // optional field for complex clauses
}
```

B.1.5 Extraction Output Examples

The following examples demonstrate the practical application of our entity extraction methodology, illustrating both implicit deontic modal detection and complex clause decomposition capabilities:

Example 1: Implicit Deontic Modal Detection

```
{
  "clause": "No other right or license,
  whether expressed or implied, in the
  Confidential Information is granted to
  the Parties hereunder.",
  "clause_type": "operative",
  "entities": {
    "Subject": "implicit grantor",
    "Deontic": "shall not grant (implicit
    prohibition)",
    "Action": "grant",
    "Object": "any other right or license in
    the Confidential Information",
    "Temporal_Condition": "not specified",
    "Contextual_Parameters": "hereunder",
    "contains_defined_terms": "Confidential
    Information, Parties",
    "counterparty": "the Parties"
  }
}
```

Example 2: Complex Clause Decomposition

```
{
  "clause": "a. The Recipient may disclose
  Confidential Information pursuant to any
  governmental, judicial, or
  administrative order, subpoena,
  discovery request, regulatory request or
  similar method, provided that the
  Recipient promptly notifies, to the
  extent practicable.",
  "clause_type": "operative",
  "entities": {
    "Subject": "The Recipient",
    "Deontic": "may (permission)",
    "Action": "disclose",
    "Object": "Confidential Information",
    "Temporal_Condition": "not specified",
    "Contextual_Parameters": "pursuant to any
    governmental, judicial, or
    administrative order, subpoena,
    discovery request, regulatory request or
    similar method, provided that the
    Recipient promptly notifies, to the
    extent practicable",
    "contains_defined_terms": "Recipient,
    Confidential Information",
    "counterparty": "not specified"
  },
  "complexity_analysis": {
    "reasoning": "Multiple obligations with
    different deontic modals: 'may disclose'
    (permission) and implicit 'must notify'
    (obligation from 'provided that').
    These represent distinct actions that
    should be processed separately for
    clarity.",
    "split_clauses": [
      "The Recipient may disclose
      Confidential Information pursuant to any
      governmental, judicial, or
```

```
administrative order, subpoena,
discovery request, regulatory request or
similar method.",
  "The Recipient must promptly notify,
  to the extent practicable."
]
}
}
```

B.2 Semantic Clustering of Legal Clauses

B.2.1 Motivation: From Flat Entities to Relational Structure

Entity extraction (Section B.1) produces flat, independently annotated clauses. Graph construction (Section B.3) requires understanding which clauses are semantically related to infer typed edges. Clustering provides this intermediate semantic organization, grouping clauses by functional relationships while preserving the entity-level detail needed for precise edge inference.

B.2.2 Methodology

Unlike document-level clustering approaches (Lu et al., 2011), our method operates at the clause level within individual contracts, where each clause may serve multiple regulatory functions simultaneously. Legal contracts are inherently multi-topical (Lu et al., 2011), requiring soft clustering that permits multi-cluster membership rather than exclusive assignments.

Our approach is **purpose-driven**: clusters are constructed to facilitate graph edge inference (Section B.3) rather than for document organization alone. The clustering process is **entity-grounded**, leveraging all extracted entity fields from Stage 1 (clause_type, Action, Object, Deontic, Temporal_Condition, contains_defined_terms) combined with derived semantic indicators (obligation_type, action_category, information_lifecycle).

The methodology follows a two-step process: (1) *systematic entity-pattern recognition* identifies clauses with matching or semantically similar entity attributes, and (2) *flexible semantic grouping* organizes these patterns into functional clusters based on three legal dimensions:

- **Functional categorization**: Grouping clauses by legal purpose based on action patterns (information disposal, protection obligations, access control, rights restrictions, definitional provisions)

- **Lifecycle stage analysis:** Temporal organization based on when obligations apply (initial disclosure requirements, ongoing protection duties, end-of-term disposal obligations)
- **Deontic role classification:** Grouping by obligation type using extracted deontic modals (mandatory obligations, prohibitions, conditional permissions)

This entity-grounded approach differs from text-similarity methods by using structured legal primitives (who-does-what-when) to identify functional relationships, enabling more precise semantic grouping for downstream graph construction.

B.2.3 Multi-Cluster Membership Framework

A key innovation is explicit support for multi-cluster membership, acknowledging that legal clauses frequently serve multiple regulatory functions simultaneously. This soft clustering approach aligns with prior work recognizing the multi-topical nature of legal documents (Lu et al., 2011). Clauses exhibiting entity patterns consistent with multiple legal functions are assigned to multiple clusters with structured justification. For example, a clause requiring "protection and restricted disclosure of Confidential Information" serves both Information Protection and Access Control functions.

B.2.4 Clustering Prompt Architecture

The clustering employs engineered prompts that balance systematic entity analysis with semantic reasoning:

Semantic Clustering Prompt

Analyze the following legal clauses and group them into clusters based on systematic entity analysis and functional relationships.

SYSTEMATIC CLUSTERING APPROACH:

STEP 1: Entity Pattern Recognition

Use the extracted entity patterns to inform clustering:

- **Clause Types:** Consider clause_type patterns (operative, definitional, exception, condition)
- **Action Categories:** Leverage action_category field (information_disposal,

information_sharing, information_protection, etc.)

- **Subject & Object Patterns:** Group clauses with related subjects, actions, and objects
- **Defined Terms:** Consider clauses sharing defined terms (contains_defined_terms field)
- **Lifecycle Stages:** Use information_lifecycle field (end_of_lifecycle, ongoing_protection, etc.)
- **Obligation Types:** Consider obligation_type (obligation, prohibition, permission)

STEP 2: Apply Flexible Clustering

Criteria 1. Semantic Similarity:

Group clauses that discuss similar legal concepts, rights, or obligations

2. Entity Overlap:

Clauses with similar subjects, actions, or objects should be grouped together

3. Functional Relationship:

Group clauses that work together to govern a specific aspect of the agreement

4. Topical Alignment:

Identify clauses addressing the same general topic area

GENERALIZABILITY FOCUS:

- Prioritize broad functional categories that would apply across contract types
- Consider whether clusters reflect generalizable legal groupings
- Avoid overly granular distinctions that wouldn't generalize to other contract types
- Think about how legal professionals would organize these clauses for practical contract management
- Avoid single-clause clusters
- merge them with the most related broader cluster

Design Rationale The prompt architecture embodies three critical design choices that distinguish our approach:

1. **Entity-basis grounding:** Explicit requirement to populate entity_basis fields with

actual extracted values (specific subjects, actions, objects) rather than generic placeholders, ensuring clustering decisions trace back to systematic entity analysis rather than subjective semantic judgments

2. **Multi-cluster membership support:** Structured framework for identifying and justifying clauses that serve multiple legal functions, validated through soft clustering approaches from prior work (Lu et al., 2011)
3. **Outlier accommodation:** Recognition that some clauses may not fit standard functional categories, preventing forced clustering that would reduce interpretative accuracy

B.2.5 Clustering Output Analysis

The clustering methodology produces structured outputs that demonstrate both systematic entity utilization and meaningful legal functional organization. Analysis of representative clustering results reveals several key patterns:

- **Functional Convergence:** Within individual documents, semantically coherent functional clusters emerge, including Information Protection Obligations, Information Disposal Requirements, Access Control Restrictions, and Confidential Information Definitions. These cluster types recur across different contracts in our dataset, suggesting the approach generalizes beyond document-specific structure.
- **Multi-Functional Recognition:** The system successfully identifies clauses serving multiple functions through the multi-cluster membership mechanism. For instance, clauses requiring "protection of Confidential Information from unauthorized disclosure" are assigned to both Information Protection and Access Control clusters.
- **Entity Pattern Utilization:** Cluster `entity_basis` fields consistently reflect the underlying entity patterns. The `common_actions` fields show semantic coherence (e.g., "protect", "hold in confidence", "maintain" clustering together for information protection), while `shared_defined_terms` reveal terminological consistency within functional groups.

Example Clustering Output

```
{
  "cluster_id": 2,
  "cluster_name": "Information Protection Obligations",
  "theme": "Clauses requiring the protection of Confidential Information during its lifecycle",
  "clause_ids": ["evidence_5", "evidence_12", "evidence_13"],
  "entity_basis": {
    "common_clause_types": ["operative"],
    "common_subjects": ["the parties", "The Receiving Party"],
    "common_actions": ["hold in confidence", "use the same degree of protection", "prevent and protect"],
    "common_objects": ["Confidential Information"],
    "shared_defined_terms": ["Confidential Information", "Receiving Party"]
  },
  "reason": "These clauses impose obligations on the parties to protect Confidential Information from unauthorized disclosure or misuse. They focus on the ongoing protection of information during the lifecycle of the agreement, with shared themes of confidentiality, security, and restricted access.",
  "confidence": "high"
}
```

Transition to Graph Construction The semantic clusters produced by this methodology serve as the foundation for edge inference in Section B.3. Intra-cluster edges typically represent *refinement* or *specification* relationships (e.g., one protection obligation detailing another), while inter-cluster edges capture cross-functional dependencies (e.g., definitional clauses linked to operative clauses using those terms). This dual structure enables the graph construction phase to leverage both entity-level precision and cluster-level semantic context.

B.3 Graph Construction via Clause Linking

B.3.1 Motivation: From Clusters to Graphs

While semantic clustering (Section B.2) identifies *which* clauses are functionally related, it does not capture *how* they relate to one another. Legal reasoning and compliance analysis require understanding the precise nature of inter-clause dependencies: Does one clause define terms used in another? Do two clauses impose complementary obligations? Does one create an exception to another's general rule? Graph construction addresses this need by inferring typed edges between clause nodes, trans-

forming flat cluster assignments into a relational structure that supports systematic legal reasoning. The resulting clause graph enables queries about definitional dependencies, obligation chains, exception hierarchies, and potential conflicts—critical capabilities for legal document understanding and question answering.

B.3.2 Relationship Taxonomy and Two-Phase Approach

Our methodology infers six core relationship types that generalize across diverse contractual domains:

Relationship Type	Description
DEFINES	One clause defines terms or concepts used in another
COMPLEMENTS	Clauses work together to govern the same legal aspect
EXCEPTIONS	One clause creates an exception to another's general rule
DEPENDS_ON	One clause requires another to be triggered or applied
CONFLICTS	Potential contradiction in legal obligations or permissions
SEQUENCE	Temporal or procedural ordering relationship

Table 9: Core relationship types for clause linking

While recent legal AI approaches like OLG++ (Dasgupta et al., 2025) create rule-level graphs for individual legal provisions using deontic logic primitives, our approach constructs document-level clause graphs that capture inter-clause relationships across entire legal documents. The two approaches operate at complementary granularities: OLG++ models fine-grained obligation structures within single rules, while our method models semantic and logical relationships between multiple clauses across a document.

Our linking methodology follows a two-phase approach that leverages the outputs of both entity extraction (Section B.1) and semantic clustering (Section B.2):

Intra-cluster linking identifies relationships within semantically coherent groups (clauses in the same cluster). **Inter-cluster linking** identifies relationships across functional groups (clauses in different clusters). This two-phase structure enables efficient analysis while maintaining interpretability: intra-cluster edges capture functional coherence, while inter-cluster edges capture cross-cutting legal logic.

B.3.3 Linking Methodology

Intra-Cluster Linking Intra-cluster linking leverages the semantic context established during clustering. Clauses within the same cluster share common entity patterns (extracted in Stage 1) and functional themes (identified in Stage 2). The linking process identifies how these semantically related clauses interact: Do they impose obligations on different aspects of the same activity? Do they specify different temporal phases of the same process? Do they provide complementary constraints? By analyzing entity overlap (shared subjects, actions, objects) and cluster context (common functional themes), the system infers relationships such as COMPLEMENTS (working together), SEQUENCE (temporal ordering), or CONFLICTS (potential contradictions).

Inter-Cluster Linking Inter-cluster linking identifies dependencies across functional boundaries. The most common pattern is definitional: a "Confidential Information Definition" cluster provides terms used by "Information Protection Obligations," "Access Control," and "Disposal Requirements" clusters. Other patterns include exception relationships (an "Exceptions to Confidentiality" cluster creates carve-outs to obligations in protection clusters) and sequential dependencies (a "Permitted Disclosures" cluster enables actions that precede obligations in a "Disposal Requirements" cluster). Inter-cluster analysis uses both entity-level information (which clusters share defined terms) and cluster-level themes (which functional categories naturally depend on one another) to infer these cross-cutting relationships.

Conservative Approach Following the evidence-based philosophy established in prior stages, our linking methodology employs high confidence thresholds and requires structured justification for each inferred edge. Relationships must be grounded in concrete entity patterns or clear cluster context. This conservative approach prioritizes precision over recall, ensuring meaningful, interpretable relationships rather than speculative connections.

Graph Visualizations Figures 5 and 6 show complete clause graphs for two NDA documents. Nodes represent individual clauses, organized spatially by cluster assignments. Edges represent typed relationships: solid lines indicate intra-cluster relationships, while dashed lines indicate

inter-cluster relationships. Edge colors correspond to relationship types (legend shown in figures). Node colors distinguish single-cluster clauses from multi-cluster clauses.

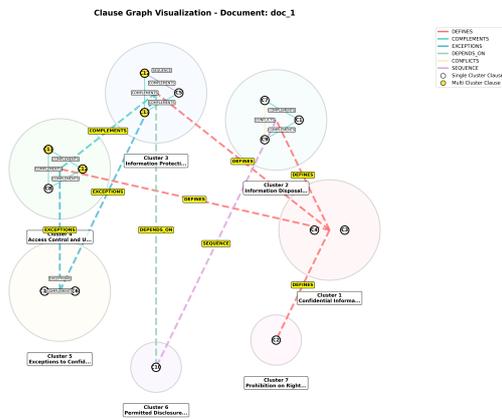


Figure 5: Clause graph for Document 1 showing 7 clusters with 12 intra-cluster and 9 inter-cluster relationships. The definitional cluster (Cluster 1) exhibits high out-degree via DEFINES edges to operative clusters. The exception cluster (Cluster 5) creates EXCEPTIONS edges to obligation clusters. The graph structure reveals the document’s logical organization: definitions establish scope, operative clauses impose obligations, and exceptions create carve-outs.

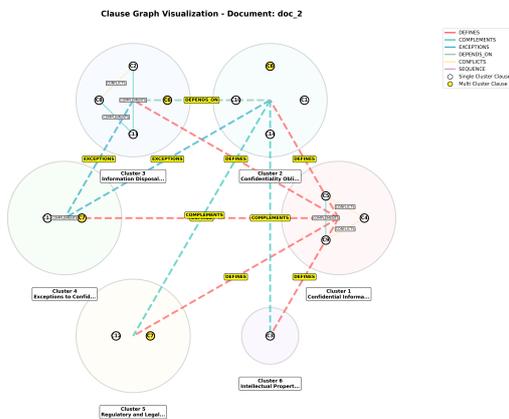


Figure 6: Clause graph for Document 2 showing 6 clusters with 8 intra-cluster and 7 inter-cluster relationships. This document exhibits a more balanced distribution of relationship types, with COMPLEMENTS relationships dominating within clusters and DEFINES relationships dominating between clusters. The visualization illustrates how different documents exhibit different structural patterns while remaining analyzable through the same framework.

B.3.4 Prompt Architecture

The linking process employs engineered prompts that integrate entity-level features from Stage 1,

cluster-level context from Stage 2, and legal domain knowledge about relationship patterns.

Intra-Cluster Linking Prompt

You are a legal expert analyzing relationships between ALL clauses within a single thematic cluster. Your task is to identify systematic legal relationships using entity patterns and cluster context.

CLUSTER CONTEXT: - cluster_name: [name] - theme: [functional theme] - entity_basis: [common subjects, actions, objects, defined terms] - reason: [why these clauses are grouped]

RELATIONSHIP TYPES TO DETECT: - DEFINES: One clause defines terms/concepts used in another - COMPLEMENTS: Clauses work together to govern the same legal aspect - EXCEPTIONS: One creates an exception to another’s general rule - DEPENDS_ON: One requires another to be triggered or applied - CONFLICTS: Potential contradiction in legal obligations/permissions - SEQUENCE: Temporal or procedural ordering relationship

CLAUSES IN THIS CLUSTER: [For each clause: clause_id, clause_type, clause_text (truncated), subject, action, object, defined_terms]

ANALYSIS GUIDELINES: 1. Use Cluster Context: Theme and entity basis provide strong hints about likely relationships 2. Definition-Usage Patterns: Look for definitional clauses whose terms are used in operative clauses 3. Complementary Actions: Different actions on same objects that work together 4. Exception Patterns: Exception clauses that create carve-outs to general rules 5. Temporal/Procedural Sequences: Based on temporal conditions or procedural ordering 6. Dependency

Relationships: Where one clause enables or requires another

CONSERVATIVE APPROACH: - Only identify relationships with CLEAR evidence from entity patterns - Focus on relationships that matter for legal interpretation - Use high confidence (0.8+) for obvious relationships, medium (0.7+) for likely ones - Avoid forcing relationships - return empty array if no clear connections exist - Each relationship must have solid reasoning based on cluster context and entities

OUTPUT FORMAT (JSON):

```
{
  "cluster_relationships": [
    {
      "source_clause_id": "clause_id_1",
      "target_clause_id": "clause_id_2",
      "relationship_type": "DEFINES|
      COMPLEMENTS|EXCEPTIONS|...",
      "confidence": 0.0-1.0,
      "explanation": "Clear explanation
      based on entity patterns",
      "evidence": {
        "entity_overlap": "specific shared
        entities/terms",
        "cluster_basis": "how cluster
        context supports this",
        "legal_significance": "why this
        matters for interpretation"
      }
    }
  ]
}
```

Inter-Cluster Linking Prompt

You are a legal expert analyzing relationships between thematic clusters in a contract document. Your task is to identify meaningful connections between clusters based on their themes, entity patterns, and legal functions.

CLUSTERS IN DOCUMENT: [For each cluster: cluster_id, cluster_name, theme, clause_types, common_actions, common_objects, shared_defined_terms]

RELATIONSHIP TYPES TO DETECT:

- **DEFINES:** Definition cluster provides terms used by other clusters - **COMPLEMENTS:** Clusters work together to govern the same legal domain - **EXCEPTIONS:** Exception cluster creates carve-outs to rules in other clusters - **DEPENDS_ON:** One cluster's effectiveness depends on another cluster - **SEQUENCE:** Clusters represent sequential steps in a legal process

ANALYSIS GUIDELINES:

1. **Definition-Usage Patterns:** - Look for definitional clusters whose shared_defined_terms appear in other clusters - These are usually the strongest inter-cluster relationships

2. **Complementary Domains:** - Clusters addressing different aspects of the same legal concern - Example: "Information Protection" + "Access Control" both govern confidentiality

3. **Exception Relationships:** - Exception clusters that create carve-outs to obligations in other clusters - Look for themes about exceptions, exclusions, or limitations

4. **Procedural Dependencies:** - Clusters representing sequential steps (e.g., "Disclosure" → "Protection" → "Return") - Based on temporal or procedural logic

CONSERVATIVE APPROACH: - Focus on clear, natural relationships based on cluster themes and entity overlap - Higher confidence threshold for cluster-level relationships (0.8+) - Each relationship must have clear legal justification - Avoid speculative connections

OUTPUT FORMAT (JSON):

```
{
  "inter_cluster_relationships": [
    {
      "source_cluster_id": 1,
      "target_cluster_id": 3,

```

```

"relationship_type": "DEFINES|
COMPLEMENTS|EXCEPTIONS|...",
"confidence": 0.0-1.0,
"explanation": "Clear explanation
based on cluster themes",
"evidence": {
  "theme_connection": "how cluster
themes relate",
  "entity_overlap": "shared terms/
entities between clusters",
  "legal_significance": "why this
relationship matters"
}
}
]
}

```

Design Rationale The prompt architecture integrates entity-level features from Stage 1, cluster-level context from Stage 2, and structured evidence requirements that mandate explicit justification for each relationship. This ensures claims trace back to textual evidence while supporting interpretability and error analysis.

B.3.5 Output Examples

Representative Linking Results The following examples illustrate typical relationships inferred by the clause linking system:

Sample Intra-Cluster Linking Output

```

{
  "source_clause_id": "evidence_1",
  "target_clause_id": "evidence_7",
  "relationship_type": "COMPLEMENTS",
  "confidence": 0.8,
  "explanation": "Both clauses impose return
obligations for Confidential
Information but address different
scenarios: immediate return vs. request-
based return. Together they ensure
comprehensive coverage.",
  "evidence": {
    "entity_overlap": "Shared: 'Confidential
Information', 'return', 'Party'",
    "cluster_basis": "Both govern
information disposal at end of lifecycle",
    "legal_significance": "Complementary
coverage of return obligations"
  },
  "detection_method": "intra_cluster_batch",
  "link_context": "intra_cluster"
}

```

Sample Inter-Cluster Linking Output 1

```

{
  "source_cluster_id": 1,

```

```

"target_cluster_id": 3,
"relationship_type": "DEFINES",
"confidence": 0.9,
"explanation": "Cluster 1 defines '
Confidential Information,' which is the
subject of protection obligations in
Cluster 3.",
"evidence": {
  "theme_connection": "Cluster 1 provides
foundational definition",
  "entity_overlap": "Shared term: '
Confidential Information'",
  "legal_significance": "Protection
obligations depend on scope defined in
Cluster 1"
},
"source_cluster_name": "Confidential
Information Definition",
"target_cluster_name": "Information
Protection Obligations"
}

```

Sample Inter-Cluster Linking Output 2

```

{
  "source_cluster_id": 5,
  "target_cluster_id": 3,
  "relationship_type": "EXCEPTIONS",
  "confidence": 0.9,
  "explanation": "Cluster 5 outlines
exceptions to confidentiality
obligations, directly limiting
obligations in Cluster 3.",
  "evidence": {
    "theme_connection": "Creates carve-outs
to protection obligations",
    "entity_overlap": "Shared term: '
Confidential Information'",
    "legal_significance": "Limits scope of
protection obligations"
  },
  "source_cluster_name": "Exceptions to
Confidentiality",
  "target_cluster_name": "Information
Protection Obligations"
}

```

The resulting graphs model inter-clause interactions across the document.

C Case Generation for Compliance Evaluation

The clause graphs constructed in Section B provide structured representations of inter-clause relationships across entire contracts. However, not all clause combinations pose equal interpretive challenges for automated compliance systems. Random sampling of clause combinations is unlikely to systematically expose the edge cases that challenge legal reasoning capabilities.

Our case generation pipeline consists of three stages: (1) **pattern-based clause sampling** identifies structurally complex clause combinations from the graphs (Section C.1), (2) **adversarial scenario generation** transforms these clause combinations into realistic compliance scenarios with controlled ambiguity (Section C.2), and (3) **LLM-based validation** filters scenarios to ensure label correctness and clause governance (Section C.3). This multi-stage approach ensures that the generated evaluation data systematically targets known sources of legal reasoning difficulty.

C.1 Pattern-Based Clause Combination Sampling

C.1.1 Motivation

Legal complexity does not arise uniformly across contracts. Specific structural patterns in contractual language—cascading definitions, overlapping temporal obligations, nested exceptions—create interpretive ambiguity that challenges both human and machine reasoning. Rather than sampling clauses randomly, we leverage the graph structure and entity attributes from Stages 1-3 to systematically identify clause combinations exhibiting known complexity patterns.

C.1.2 Complexity Pattern Taxonomy

Building on legal compliance literature that identifies temporal obligations (Governatori, 2010; Giordano et al., 2013) as sources of interpretive difficulty, and motivated by challenges observed in ContractNLI (Koreeda and Manning, 2021)—including negation by exception and definitional references—we define eight structural patterns where legal ambiguity arises. These patterns challenge automated systems because they require: (1) multi-hop reasoning across distant clause boundaries, (2) disambiguation of vague terms in context, and (3) reconciliation of potentially conflicting obligations.

- **Definitional Chains:** Cascading definitions where Term A defines B defines C, creating propagating ambiguity through dependency chains
- **Temporal Conflicts:** Overlapping temporal obligations with unclear precedence, creating scheduling or deadline interpretation challenges
- **Scope Boundary Disputes:** Overlapping permission and restriction domains where boundaries between allowed and prohibited actions are unclear

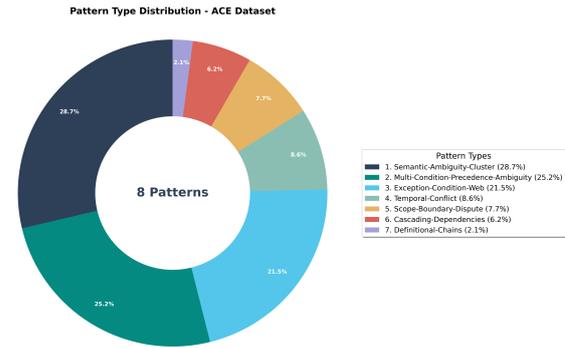


Figure 7: Distribution of scenario pattern types in the ACE Dataset.

- **Cascading Dependencies:** Chain of dependent obligations where failure at any point affects downstream obligations
- **Semantic Ambiguity Clusters:** Multiple clauses relying on the same vague terms without shared interpretation standards
- **Exception-Condition Webs:** Multiple conditions or exceptions creating complex interaction between general rules and carve-outs
- **Authority Conflicts:** Different actors granted authority over overlapping decision domains
- **Multi-Condition Precedence:** Single clause with multiple conditional elements lacking clear evaluation order

Figure 7 shows the distribution of these pattern types in the ACE dataset.

C.1.3 Pattern Detection and Scoring

For each pattern type, detection operates on graph structures and entity attributes:

- **Definitional Chains:** Traverse DEFINES edges to identify chains where `contains_defined_terms` in clause i matches `defined_term` in clause $i - 1$
- **Temporal Conflicts:** Match clauses with overlapping temporal keywords in `Temporal_Condition` or conflicting numeric deadlines
- **Scope Boundary Disputes:** Identify permission/prohibition pairs with Jaccard similarity > 0.7 on Object and Subject entities
- **Cascading Dependencies:** Find `DEPENDS_ON` chains of length ≥ 3 across graph edges

- **Semantic Ambiguity:** Group clauses sharing vague terms from curated list (e.g., "reasonable," "material")
- **Exception-Condition Webs:** Detect EXCEPTIONS edges with multiple conditions via complexity_analysis field
- **Authority Conflicts:** Match authority keywords with overlapping scope
- **Multi-Condition Precedence:** Extract clauses where $|\text{split_clauses}| \geq 4$, excluding definitional types

Example Patterns. Table 10 illustrates two representative patterns detected by the sampling process.

Pattern Type	Detected Instance
Multi-Condition Precedence (Score: 10.3)	<i>Clause:</i> "Party shall return all Confidential Information... except that portion found in analyses... and Representatives shall not retain copies except Party may retain one copy as needed to comply with applicable law..." <i>Justification:</i> 5 conditional elements with unclear precedence (return obligation, analysis exception, no-retention rule, legal compliance exception).
Semantic Ambiguity Cluster (Score: 9.7)	<i>Vague term:</i> "including" <i>3 clauses span:</i> (1) Return obligations "including copies converted to computerized media", (2) Definition "including but not limited to: source code, technical design...", (3) Scope "including but not limited to:" <i>Justification:</i> Non-exhaustive list marker creates interpretive ambiguity across obligation, definition, and scope clauses.

Table 10: Representative complexity patterns detected by the sampling process.

Enhanced Complexity Scoring. Beyond rule-based pattern detection, we incorporate graph-theoretic metrics—relationship type entropy, cross-cluster edge density, and network density—to quantify structural complexity (Newman, 2010).

$$\text{Score}_{\text{enh}} = (\text{Score}_{\text{base}} + w_e H + w_c D_{\text{cross}} + w_d D_{\text{net}}) \cdot \phi_{\text{conf}} \quad (2)$$

where $\text{Score}_{\text{base}}$ is the rule-based pattern score, H is relationship type entropy ($H = -\sum_r p_r \log_2 p_r$), D_{cross} is cross-cluster edge density, D_{net} is network density, and ϕ_{conf} is average extraction confidence from Stage 1. Weights ($w_e = 0.5$, $w_c = 2.0$, $w_d = 1.5$), pattern

detection thresholds (Jaccard similarity 0.7), and confidence filtering (0.7) are empirically determined. Cross-cluster patterns receive bonuses (+3 to +4) as they indicate system-wide complexity. The sampling process outputs ranked clause combinations with their complexity scores, clause texts, and justifications, which serve as input to scenario generation (Section C.2).

C.2 Adversarial Scenario Generation

C.2.1 Design Rationale: Avoiding Synthetic Distribution Shift

Prior work on contextual integrity-based synthetic data generation exhibits a systematic performance gap between synthetic and real evaluation cases. Li et al. (2025a) report that models achieve 92.64% average accuracy on synthetic HIPAA cases but only 80.29% on real court cases—a 12% gap indicating that synthetic scenarios contain detectable distributional artifacts that inflate model performance.

Manual analysis reveals two primary sources of these artifacts in existing synthetic datasets: (1) **explicit feature enumeration**, where narratives directly state CI characteristics (“the covered entity sends protected health information to the patient for treatment”), and (2) **compliance-indicative tonality**, where legal compliance is signaled through procedural language rather than neutral narrative description. Table 11 contrasts a representative synthetic case from Fan et al. (2024) with a real HIPAA court case, illustrating these artifacts.

Source	Case Narrative Excerpt
GoldCoin Synthetic	“Jane ... visited ... physician, Dr. Smith ... During the appointment, Dr. Smith discovered abnormalities in her blood test results and sent the results to Dr. Adams, for specialist diagnostic assessment and treatment planning.”
Real HIPAA Case	“In Chester County, an 88-year-old woman, A.M., found herself at the center of a complex legal situation. The county’s Department ... received a confidential report ... Bernard ... understood the gravity ... His role was to gather information ... to determine if protective services were necessary ...”

Table 11: Comparison of synthetic and real case narratives. The synthetic case explicitly states purpose (“treatment planning”) and uses procedural framing, while the real case presents a neutral narrative requiring inference about legal obligations.

To address this distribution shift, our scenario generation methodology employs adversarial

prompting strategies designed to create scenarios that are *technically* compliant or non-compliant while *appearing* otherwise to surface-level analysis. This approach forces evaluation systems to perform multi-hop legal reasoning rather than pattern matching on synthetic artifacts.

C.2.2 Scenario Generation Strategy

Our pipeline generates three scenario types:

Compliant with Distractor. Scenarios where actions strictly satisfy regulatory requirements but are masked by misleading contextual details (e.g., informal communication channels, minimalist compliance approaches, or letter-versus-spirit conflicts).

Violation with Plausible Defense. Scenarios containing clear regulatory breaches masked by superficially reasonable business justifications (e.g., good-faith intent, scope misapplication, or de minimis reasoning).

Non-Applicable via Cross-Matching. For non-applicable scenarios, we employ within-document cross-matching rather than random negatives. For each document, we identify all clause combination patterns from Stage 1. We then pair clause combination C_i with scenarios originally generated for non-overlapping combination C_j from the same document, where $C_i \cap C_j = \emptyset$. This ensures non-applicable scenarios remain privacy-relevant while being governed by different regulatory provisions, avoiding trivial negatives.

C.2.3 Generation Prompts

The following prompts implement our adversarial generation strategy for compliant and violation scenarios.

Compliant Scenario Generation Prompt

You are a specialist in Adversarial Testing for Legal AI. Your mission is to design a scenario that is **technically compliant** but appears questionable at first glance.

MISSION DIRECTIVE:

1. **Construct a "Technical Compliance" Scenario:** The party's actions must strictly adhere to the letter of the clauses.

2. **Introduce a "Plausible Distractor":** Weave a misleading fact or seemingly unprofessional action that makes the party's actions appear non-compliant to superficial analysis.

3. **Create a Path to Correct Reasoning:** The scenario must contain all necessary facts for a sophisticated AI to logically conclude compliance by connecting actions to specific clause language.

EXAMPLE TECHNIQUES FOR DISTRACTOR:

- **Informal Communication:** Fulfill formal requirements through unconventional channels (e.g., deleted chat message)

- **"Letter vs. Spirit" Conflict:** Meet exact wording but seem to violate intuitive policy intent

- **Minimalist Action:** Do absolute bare minimum required, appearing negligent but technically sufficient

REQUIRED JSON OUTPUT:

```
{
  "compliance_label": "Compliant",
  "scenario": "Detailed narrative with
              technical compliance
              masked by plausible
              distractor...",
  "question": "Did [Party] act in
              compliance when...",
  "distractor_element": "Description of
              misleading fact...",
  "path_to_correct_reasoning": "Legal
                              analysis explaining
                              how to
                              overcome distractor..."
}
```

Violation Scenario Generation Prompt

You are a specialist in Adversarial Testing for Legal AI. Your mission is to design a scenario that contains a **clear violation** masked by a superficially reasonable justification.

MISSION DIRECTIVE:

1. **Construct a "Deceptive Violation" Scenario:** The party's actions must represent a definitive breach of the clauses.
2. **Introduce a "Plausible Defense":** Create a compelling but ultimately invalid justification. The party should act in ways that seem reasonable but technically violate the agreement.
3. **Create a Path to Correct Reasoning:** The scenario must contain facts needed to logically invalidate the plausible defense by correctly applying clause language.

EXAMPLE TECHNIQUES FOR PLAUSIBLE DEFENSE:

- **"Good Intentions" Breach:** Violate clause (e.g., confidentiality) with stated goal of helping, but without proper authorization
- **Misinterpreted Scope:** Apply permission from one clause to situation governed by different, more restrictive clause
- **De Minimis Violation:** Minor financial breach but clear violation of strict procedural rule

REQUIRED JSON OUTPUT:

```
{
  "compliance_label": "Non-Compliant",
  "scenario": "Detailed narrative where
    action seems reasonable
    but constitutes clear
    breach...",
  "question": "Did [Party] breach
    obligations when...",
  "plausible_defense": "Seemingly
    reasonable justification...",
  "path_to_correct_reasoning": "Legal
    analysis explaining
    why
    defense fails..."
}
```

C.3 LLM Based Validation

C.3.1 Why Validation is Critical

Generated cases must accurately reflect legal requirements from source documents. Traditional metrics like BLEU and ROUGE are inadequate

for assessing legal accuracy (Badshah and Sajjad, 2024). Manual validation at scale is resource-prohibitive. We require a scalable yet reliable validation approach.

C.3.2 LLM-as-Judge Approach

Why use LLM-based validation: LLMs have demonstrated strong correlation with human judgment in evaluation tasks (Verga et al., 2024; Badshah and Sajjad, 2024), offering a scalable alternative while maintaining reliability.

Critical challenge: Research shows that LLMs exhibit self-preference bias, rating their own outputs higher than others' even when humans judge them as equal quality (Panickssery et al., 2024). This bias poses a significant threat to validation objectivity.

Our mitigation strategy: We employ Claude Sonnet 4 as the validator, distinct from our generation models (GPT-4o). Using evaluators from different model families has been shown to reduce intra-model bias (Verga et al., 2024), providing more independent judgment.

C.3.3 Reference-Guided Evaluation

We adopt a reference-guided validation approach where the validator receives: (1) the generated case scenario, (2) the original regulation excerpt (governing clauses) serving as ground truth, (3) the claimed compliance label, and (4) the provided reasoning. This approach is critical because reference answers have been shown to reduce self-enhancement bias, enabling models to objectively judge outputs even from their own model family (Badshah and Sajjad, 2024).

Rather than asking "is this case good?" our validation framework asks two specific questions: "is this scenario meaningfully governed by the provided clauses?" and "is the compliance label correct given the clauses?"—fundamental requirements in the legal domain where fidelity to source material is paramount.

C.3.4 Validation Dimensions

Our validation framework evaluates two dimensions, each addressing a distinct quality requirement:

1. **Clause Governance:** Determines whether the central legal conflict in the scenario is directly and meaningfully governed by the provided regulatory clauses. A scenario passes this check if the legal question at its core can only

be answered by applying the specific rules in these clauses, rather than being tangentially related or describing general business issues. This is scored via a binary decision (`is_governed_by_clauses`) and a relevance score (0-1 scale).

2. **Label Correctness:** Independently verifies whether the assigned compliance label accurately reflects the application of governing clauses to the scenario facts. The validator forms its own legal conclusion and critically evaluates the provided reasoning to determine if it constitutes a logically sound argument. This is scored via a binary decision (`is_label_correct`) and a confidence score (0-1 scale).

Each dimension produces a binary decision with accompanying justification and a continuous score reflecting the evaluator's confidence/relevance assessment. The justifications provide interpretability for downstream improvements and create an audit trail for legal review.

Cases must satisfy both validation checks to be accepted: (1) clause governance with relevance score ≥ 0.7 , and (2) label correctness with confidence score ≥ 0.8 . Cases failing either check are rejected. Cases passing both checks but with scores below the thresholds are flagged for manual review, enabling quality control while maintaining efficiency.

The following prompt implement our LLM based scenario validation.

LLM Based Validation Prompt

You are a meticulous and impartial legal adjudicator. Your task is to critically evaluate a legal scenario against a set of governing clauses. Analyze the provided information and return a strict JSON object with your findings.

//- CASE FILE -//

1. **GOVERNING CLAUSES (The Ground Truth):**

[Evidence texts in JSON format]

2. **SCENARIO TEXT:**

[Generated scenario text]

3. **CLAIMED COMPLIANCE LABEL (The**

Claim):

[Compliance label]

4. **PROVIDED REASONING (The Argument to Test):**

[Path to correct reasoning]

//- ADJUDICATION DIRECTIVE -//

Follow these two steps in your analysis:

Step 1: Assess Clause Governance

Determine if the central conflict described in the SCENARIO TEXT is directly and meaningfully governed by the GOVERNING CLAUSES.

- A scenario **IS GOVERNED** if the legal question at its heart can only be answered by applying the specific rules in these clauses.
- A scenario **IS NOT GOVERNED** if the clauses are only tangentially related or if the scenario describes a general business issue that doesn't hinge on these specific rules.

Step 2: Assess Label Correctness

Independently determine if the CLAIMED COMPLIANCE LABEL is correct.

- First, form your own conclusion by applying the GOVERNING CLAUSES to the SCENARIO TEXT.
- Then, evaluate the PROVIDED REASONING. Is it a logically sound argument that correctly applies the clauses to the facts to justify the CLAIMED COMPLIANCE LABEL? Your final decision on label correctness must be based on this critical evaluation.

//- REQUIRED JSON OUTPUT -//

Based on your analysis, provide your judgment in the following JSON format. **Do not add any text outside of the JSON object.**

```
{
  "clause_governance_check": {
    "is_governed_by_clauses": true,
    "justification": "The scenario's core
issue regarding
notification"
```

```

timelines is directly
    addressed by clause
7.2(a).",
"relevance_score": 0.9
},
"label_correctness_check": {
"is_label_correct": false,
"justification": "The label is
incorrect. The scenario
states the party
acted on the 5th
business day, but
clause 3.1 specifies
'five calendar days'
, making their
action non-compliant
.",
"confidence_score": 1.0
}
}

```

C.4 Model Training

Each instance of validated reasoning label pairs is formatted in an instruction-following conversational format, containing a system prompt that defines the model’s analytical role and an assistant response structured as:

```

<reasoning> step-by-step justification ...
</reasoning>
<final_solution> Compliant / Non-Compliant /
Not-Applicable </final_solution>

```

C.4.1 Training Configuration

To make training efficient while maintaining fidelity to the teacher’s reasoning process, we adopt **LoRA (Low-Rank Adaptation)** for parameter-efficient fine-tuning, updating only selected projection layers.

The models were trained using the following setup:

- **LoRA configuration:**
 - Rank (r) = 16
 - Scaling factor (α) = 32
 - Dropout = 0.05
 - Target modules = [q_proj, k_proj, v_proj, o_proj]
- **Training settings:**
 - Maximum sequence length = 8,192
 - Per-device batch size = 1 (for both training and evaluation)
 - Gradient accumulation steps = 1
 - Number of epochs = 3

- Learning rate = 1×10^{-4}
- Weight decay = 0.01
- Warmup ratio = 0.1

D Compliance Evaluation Task

To establish a baseline for performance, all models were evaluated in a zero-shot setting using a standardized prompt format. This prompt is designed to be domain-agnostic, providing the model with the necessary legal context (regulatory provisions) and a factual scenario, then requesting classification. This approach ensures a fair comparison across different models and datasets (ACE, EU AI Act, HIPAA).

Compliance Evaluation Prompt

You are a legal expert analyzing regulatory compliance cases. Given a case scenario and relevant regulatory provisions, determine whether the scenario represents a VIOLATION, COMPLIANT, or NOT APPLICABLE under the applicable regulations.

Definitions:

- **VIOLATION:** The case scenario describes actions that are non-compliant with or explicitly forbidden by the provided regulations.
- **COMPLIANT:** The case scenario describes actions that adhere to or are explicitly allowed by the provided regulations.
- **NOT APPLICABLE:** The case scenario describes actions that are not clearly addressed by the provided regulations, or where the regulations are ambiguous or do not apply.

Regulatory Provisions:

{relevant_articles}

Case Scenario:

{case_content}

Instructions:

- You MUST end your response with EXACTLY one of these formats:
- FINAL_ANSWER: A (for VIOLATION)

- FINAL_ANSWER: B (for COMPLIANT)
- FINAL_ANSWER: C (for NOT APPLICABLE)

Base Model: COMPLIANT

E Case Study on Model Performance

This section presents representative cases from the EU AI Act dataset where the base model fails while our fine-tuned model (Qwen2.5-3B Instruct) correctly identifies compliance status, demonstrating the effectiveness of ACE training. For brevity, we omit the referenced articles for these scenarios. We include the reasoning traces from both models to illustrate how fine-tuning improves legal reasoning capabilities.

E.1 Example 1: Non-Compliant Case

Case Scenario (Non-Compliant)

Scenario:

The AI system, named 'SafeGuard AI', is developed by a product manufacturer specializing in personal protective equipment, cableway installations, machinery, toys, and radio equipment. SafeGuard AI is designed to enhance the safety and efficiency of machinery operations by monitoring and analyzing real-time data from various equipment. It uses advanced algorithms to predict potential malfunctions and provide maintenance alerts. The system collects data from machinery sensors, including operational status, location, and usage patterns. SafeGuard AI is integrated into a network of cableway installations and radio equipment to ensure seamless communication and data transmission. The AI system also interfaces with personal protective equipment to monitor the safety conditions of workers in real-time. However, the integration of SafeGuard AI raises concerns about the privacy and security of the data collected, especially regarding the personal information of workers using the equipment. The system's ability to track and analyze individual usage patterns and locations necessitates a careful examination of data protection and consent requirements.

Ground Truth: NON-COMPLIANT

Fine-tuned Model: NON-COMPLIANT

Base Model Prediction:

Base Model Reasoning:

The scenario describes the development and deployment of an AI system, SafeGuard AI, which is designed to enhance the safety and efficiency of machinery operations. According to the EU Artificial Intelligence Act, the regulation applies to providers and deployers of AI systems that have their place of establishment or are located within the Union, or where the output produced by the AI system is used in the Union (Sections 1.a through 1.c). Since the AI system is integrated into machinery and equipment used in the Union, it falls under the scope of the regulation.

*[Critical Error:] However, the specific actions described in the scenario, such as the development, deployment, and integration of the AI system, are not explicitly prohibited or permitted by the regulation. Instead, the regulation sets out requirements for providers and deployers of AI systems, particularly regarding transparency, documentation, and risk management for high-risk AI systems. **The scenario does not indicate that these actions are explicitly prohibited, but rather that they must comply with the applicable provisions of the regulation.***

[Incorrect Conclusion:] Therefore, the actions described in the scenario are PERMITTED under the EU Artificial Intelligence Act, provided that they comply with the relevant requirements set forth in the regulation.

Key Error: The base model fails to recognize that the scenario describes *concerns about privacy and data protection violations* when collecting personal information from workers without proper consent mechanisms, which constitutes non-compliance with data protection requirements referenced in Section 7 of the regulation.

Fine-tuned Model Prediction:

Fine-tuned Model Reasoning (Excerpt):

First, I need to analyze the case scenario against the regulatory provisions provided. The scenario describes an AI system called "SafeGuard AI" developed by a product manufacturer. This system is integrated into machinery, cableway installations, radio equipment, and personal protective equipment. It monitors and analyzes real-time data to predict malfunctions and provide maintenance alerts. **Importantly, it collects data from machinery sensors and personal protective equipment, including operational status, location, and usage patterns. There are concerns about privacy and security, particularly regarding the personal information of workers.**

[...] Looking at the regulatory provisions, **Section 7 is crucial because it ties data protection to the regulation. Since the scenario involves processing personal data, and the regulation explicitly references data protection laws, this should be covered.** The scenario highlights issues with data collection and privacy, which are addressed by Section 7. Therefore, since the regulation requires adherence to data protection laws, and there are concerns about privacy, it suggests that the action—collecting and processing personal data—is subject to the regulation.

The scenario describes an action (data collection and processing) that is subject to the regulation via data protection laws, and there are privacy concerns, indicating non-compliance. Therefore, the label should be Non-Compliant.

E.2 Example 2: Compliant Case

Case Scenario (Compliant)

Scenario:

The AI system, named 'SecureGuard', is a sophisticated AI tool developed by a European-based distributor. It is designed for military purposes and is exclusively used by public authorities in third countries for law enforcement and judicial cooperation.

SecureGuard is capable of processing large volumes of data to identify potential security threats and assist in strategic decision-making. The system is not used for any commercial purposes and is strictly regulated to ensure compliance with international security protocols. Despite its military focus, SecureGuard's deployment involves the transmission of sensitive personal data, raising concerns about privacy and data protection. The system operates without direct interaction with individuals, and its data processing activities are conducted under the oversight of international organizations.

Ground Truth: COMPLIANT

Fine-tuned Model: COMPLIANT

Base Model: NON-COMPLIANT

Base Model Prediction:

Base Model Reasoning:

The scenario describes an AI system, SecureGuard, that is designed for military purposes and is exclusively used by public authorities in third countries for law enforcement and judicial cooperation. According to the EU Artificial Intelligence Act, Article 2, Section 4, this Regulation does not apply to public authorities in a third country or to international organisations falling within the scope of this Regulation, where those authorities or organisations use AI systems in the framework of international cooperation or agreements for law enforcement and judicial cooperation with the Union or with one or more Member States, provided that such a third country or international organisation provides adequate safeguards with respect to the protection of fundamental rights and freedoms of individuals.

[Critical Error:] However, the scenario also mentions that SecureGuard processes sensitive personal data, which raises concerns about privacy and data protection. While the regulation does not apply to the use of the AI system itself, it does not exempt the system from compliance with other applicable Union laws, such as those related to

the protection of personal data and privacy. Therefore, the processing of sensitive personal data by SecureGuard would need to comply with the General Data Protection Regulation (GDPR) and other relevant data protection laws, even though the AI system itself is not subject to the specific provisions of the EU Artificial Intelligence Act.

[Incorrect Conclusion:] Given that the scenario describes actions that are explicitly forbidden/non-compliant under the GDPR and other data protection laws, the action represented by the scenario is PROHIBITED under the applicable regulations.

Key Error: The base model incorrectly conflates the EU AI Act with GDPR requirements and fails to recognize that Section 3 explicitly excludes AI systems used *exclusively for military, defense, or national security purposes* from the regulation's scope, making the system compliant with the AI Act regardless of separate data protection concerns.

cluded from the regulation under Section 3. The primary exclusion is in Section 3, which explicitly allows the use for national security purposes.

The scenario mentions "raising concerns about privacy and data protection," but **the regulation itself doesn't require compliance for these purposes when the use is for national security. The regulation is silent on data protection for such systems, as per Section 3.** Therefore, SecureGuard's deployment is compliant with the regulation because it falls under the exclusion.

Fine-tuned Model Prediction:

Fine-tuned Model Reasoning (Excerpt): Looking at the regulatory provisions, **Section 3 is very relevant: "This Regulation does not apply to areas outside the scope of Union law, and shall not affect the competences of the Member States concerning national security... This Regulation does not apply to AI systems where and in so far they are placed on the market, put into service, or used with or without modification exclusively for military, defence or national security purposes..."**

In the scenario:

- SecureGuard is used by public authorities in third countries for law enforcement and judicial cooperation, which fall under national security.
- It is not placed on the market or put into service in the EU; it's used in third countries.
- The data processing is done under international organization oversight.

Therefore, SecureGuard should be ex-