Multi-Agent Discovery of Missing Legal Conflicts

Anonymous ACL submission

Abstract

Human annotations are often treated as ground truth in machine learning, but in non-verifiable domains such as law, they may reflect only one reasonable judgment among multiple valid perspectives. We study this problem in legal conflict review under California Accessory Dwelling Unit (ADU) law, where determining whether a local ordinance conflicts with state law can depend on statutory interpretation, factual context, and policy priorities. Evaluating systems against a single label may therefore mistake reasonable disagreement for model error and obscure conflicts missed by prior reviewers. We introduce PLEA (**P**arallel **L**egal **E**volving **A**gents), a multi-agent framework that instantiates legally grounded perspectives to review state–local conflict pairs. Varying the vote cutoff yields precision–recall tradeoffs that expose contested cases and prioritize candidate omissions for human review. On held-out evaluation, optimized PLEA improves over single-agent and non-optimized multi-agent baselines; expert-assisted revalidation confirms 20 of 30 sampled candidate omissions (66.7%) as missed legal conflicts. We will release code and discovered missing conflicts at `anonymous`.

1 Introduction

Machine learning and natural language processing have largely been built around tasks where each data item is assumed to have a correct label (Wang et al., 2018; Dagan et al., 2005; Kotsiantis, 2007). This assumption fits many settings: a math problem has a verifiable solution (Cobbe et al., 2021; Guo et al., 2025), the correctness of a code program can often be checked through execution (Chen et al., 2021; Austin et al., 2021), and an image can be mapped to a class label in benchmarks (Deng et al., 2009; Russakovsky et al., 2015). Even when labels are collected from humans, standard supervised-learning pipelines often collapse disagreement into a single target label through majority vote or other aggregations, treating variation primarily as noise rather than as a source of valid alternative judgments (Dawid and Skene, 1979; Snow et al., 2008; Davani et al., 2022; Gordon et al., 2022).

Yet many important language tasks are not fully verifiable in this sense. In natural language inference (Pavlick and Kwiatkowski, 2019), veridicality assessment (de Marneffe et al., 2012), and information retrieval (Demeester et al., 2016; Voorhees, 2000), multiple answers may be reasonable because the task is ambiguous, underspecified, or assessor-dependent. Prior work has shown that human disagreement in such tasks can be systematic rather than merely noisy, suggesting that evaluation should sometimes model a distribution of plausible judgments rather than collapse all annotations into a single ground truth (de Marneffe et al., 2012; Pavlick and Kwiatkowski, 2019).

This problem is especially important in legal and policy domains, where computational systems are increasingly used to support legal analysis and decision-making, but should also help institutions identify issues that past human review may have missed rather than merely reproduce prior decisions (Surden, 2014; Frankenreiter and Livermore, 2020; Kleinberg et al., 2018). We study this challenge through legal conflict detection in California Accessory Dwelling Unit (ADU) law. California's Department of Housing and Community Development (HCD) reviews local ADU ordinances to identify provisions that may be preempted because they impose restrictions inconsistent with state law. These reviews are important for housing policy, since overly restrictive local rules can limit ADU construction. A natural ML formulation would treat past HCD review letters as labeled data and evaluate systems by how well they recover previously identified conflicts. However, these annotations may reflect only the conflicts noticed under particular review practices, time constraints, and interpretive choices. A system trained or evalu-

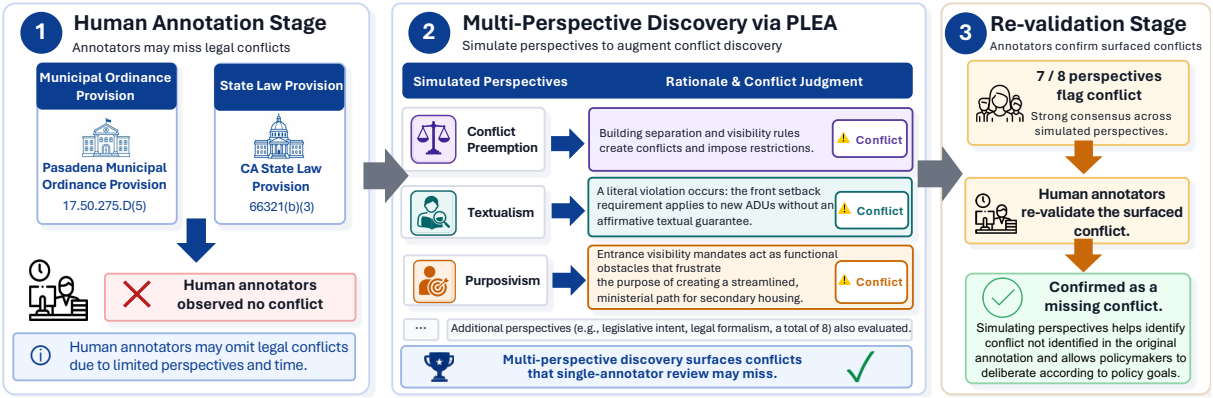

Figure 1: Overview of our workflow. PLEA simulates multiple legal perspectives to review legal texts, discovering missing legal conflicts and revalidating the findings with human annotators.

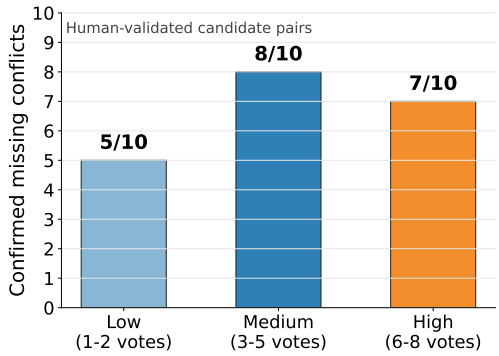

Figure 2: Human revalidation of candidate missing conflicts proposed by PLEA. We sample 30 originally negative local ordinance–state law pairs, stratified by pair-specific vote agreement, with 66.7% discovered missing conflicts. Bars report how many candidates in each vote region are confirmed by human legal review as missed conflicts. Validated omissions appear even in low-agreement cases, while medium- and high-agreement cases show stronger confirmation rates.

ated only against these labels may therefore miss genuine conflicts absent from the original record.

We propose PLEA (**P**arallel **L**egal **E**volving **A**gents), an LLM-driven multi-agent framework for discovering legal conflicts beyond existing annotations (Figure 1). Instead of treating a single annotation set as exhaustive ground truth, PLEA operationalizes legal disagreement through multiple agents, each instantiated with a distinct interpretive stance, search strategy, and tool-calling pattern for reviewing local ordinances against state law. The agents generate candidate conflicts absent from the original annotations, which are then revalidated by human annotators. In doing so, PLEA reframes evaluation from measuring agreement with past labels to auditing and improving the label set itself. Existing annotations may reflect partial deliberation, since reviewers often cannot consider every relevant interpretive stance or policy concern before recording a judgment (Farnsworth et al., 2010). By decomposing disagreement into explicit perspectives, PLEA broadens the considerations available for later human deliberation and helps policymakers decide which conflicts to prioritize, where to allocate limited review resources, and when additional legal analysis is warranted.

Empirically, PLEA improves held-out conflict detection over both a single legal reviewer and a non-optimized multi-agent baseline. More importantly, human revalidation shows that many apparent false positives are genuine annotation omissions: among 30 candidate omissions sampled across vote-agreement regions, expert-assisted review confirms 20 pairs, or 66.7%, as missed legal conflicts (Figure 2). These validated omissions appear across multiple issue types, including permitting timelines, objective design and development standards, owner-occupancy and rental restrictions, state-law exemptions, and unit-combination rules. This suggests that PLEA recovers legally meaningful omissions rather than merely generating superficial disagreement with the label set.

Our results point to a broader role for multi-agent systems in non-verifiable domains. In settings such as law, policy, search, and expert review, a model output can be useful even when it disagrees with an existing annotation, because the annotation itself may be incomplete. Multi-agent frameworks offer a way to make this incompleteness visible: they can represent competing but valid perspectives, generate alternative candidates, and support human experts in deciding which disagreements reveal real omissions. We therefore argue that, for

Perspective	Review focus
Conflict preemption	Broad recall pass over state-law hooks, omissions, and procedural effects.
Pragmatism	Practical-burden pass for cost, delay, feasibility, and administrability.
Textualism	Literal side-by-side comparison of local and state text.
Purposivism	Review for rules that undermine the purpose of ADU production and affordability.
Legislative intent	Bill-history review tied to a specific legislative choice.
Legal formalism	Doctrinal-category review for floors, ceilings, and procedures.
Agency deference	Generic admin-guidance review for ambiguous statutory applications and implementation practice.
Jurisdiction defense	Protect local interests and vote for conflict only when reasonable defenses fail.

Table 1: Compact summary of the eight legal perspectives instantiated in PLEA. Full perspective descriptions and tool-use patterns are shown in Table 3.

non-verifiable tasks, evaluation should not only ask whether models reproduce existing labels, but also whether they help improve the human label set and reveal the limits of the assumed ground truth.

2 Approach

2.1 Preliminaries

Problem Formulation Let $x \in \mathcal{X}$ denote a local ordinance provision and $s \in \mathcal{S}$ denote a related state-law provision. The task is to determine whether x conflicts with s, e.g. when the local ordinance imposes a restriction that is inconsistent with a state law. We represent a legal perspective as a function

$$f : \mathcal{X} \times \mathcal{S} \to \{0, 1\},$$

where $f(x, s) = 1$ indicates that the perspective finds a conflict, and $f(x, s) = 0$ otherwise.

Legal conflict review is often interpretive rather than deterministic and objective. Different valid legal perspectives may reach different views on the same pair (x, s). Let F denote a conceptual set of admissible legal perspectives, with $f_1, \ldots, f_m \in F$ and, in general, $f_i \not\equiv f_j$ for some $i \neq j$. For example, a textualist perspective may find a conflict by focusing on the ordinary meaning of the statutory text, while a purposivist may disagree and cite the statutory purpose and policy context.

In practice, policymakers and legal practitioners do not simply apply one fixed perspective. They deliberate across competing interpretations, institutional goals, and policy priorities. We denote this deliberative target by

$$\bar{f}(x, s) = A(F, x, s),$$

where A represents an idealized aggregation or deliberation process over admissible perspectives. We do not assume that $\bar{f}$ is directly observable. Rather, it represents the decision that would be reached after sufficiently broad and careful consideration of the relevant legal perspectives.

Real-world annotations are more limited. Due to constraints on time, expertise, and institutional resources, human reviewers may consider only a subset of relevant perspectives when annotating examples. We observe annotation triples (x_i, s_i, y_i), where $y_i \in \{0, 1\}$ is the conflict judgment recorded by human reviewers. Ideally, $y_i = \bar{f}(x_i, s_i)$, but in practice some conflicts may be under-discovered when the annotation process does not consider the full range of relevant interpretations.

Automated Review System A common automated review system uses an LLM as a judge. Given a local ordinance x, a state law s, the LLM produces a predicted conflict judgment

$$\hat{y} \sim g(\cdot \mid x, s),$$

where $\hat{y} \in \{0, 1\}$. Here, g denotes the LLM-based review procedure, including the model, prompt, context, tool calling, and decoding strategy.

Naive Evaluation A standard evaluation compares the system prediction $\hat{y}_i$ against the observed human annotation y_i. For example, the number of true positives is

$$TP = \sum_{i=1}^{n} \mathbb{1}[\hat{y}_i = 1 \wedge y_i = 1].$$

Precision and recall can then be computed as

$$\text{Precision} = \frac{\sum_{i=1}^{n} \mathbb{1}[\hat{y}_i = 1 \wedge y_i = 1]}{\sum_{i=1}^{n} \mathbb{1}[\hat{y}_i = 1]},$$

and

$$\text{Recall} = \frac{\sum_{i=1}^{n} \mathbb{1}[\hat{y}_i = 1 \wedge y_i = 1]}{\sum_{i=1}^{n} \mathbb{1}[y_i = 1]}.$$

This evaluation treats y_i as ground truth. However, if the annotation process under-discovers conflicts, then y_i may be an incomplete proxy for the

Input excerpts

Local provision: 19.07.090.C

```
Any application for an accessory dwelling unit or junior
↪  accessory dwelling unit shall include a site plan, floor
↪  plan, and elevations substantiating and evidencing
↪  compliance with all applicable development standards.

Where all requirements of this section and the Commerce
↪  Municipal Code appear to be met, the application shall
↪  be approved ministerially without discretionary review
↪  or public hearing within sixty days of receiving the
↪  application.

The city and applicant may agree to additional time with a
↪  written request from the applicant.
```

State provisions: Gov. Code §§66317(a), (b)

```
A permit application for an accessory dwelling unit or a
↪  junior accessory dwelling unit shall be considered and
↪  approved ministerially without discretionary review or a
↪  hearing [...]

The permitting agency shall either approve or deny the
↪  application ... within 60 days from the date the
↪  permitting agency receives a completed application.

If a permitting agency denies an application ... the
↪  permitting agency shall ... return in writing a full set
↪  of comments ... and a description of how the application
↪  can be remedied by the applicant.
```

PLEA output: candidate missing conflicts

Candidate conflict	Voting perspectives	Representative perspective rationales
Broad code-compliance condition	conflict preemption; pragmatism; textualism; purposivism; legislative intent; legal formalism; agency deference	*Conflict preemption:* Approval depends on whether all requirements of the ADU section and the Commerce Municipal Code "appear to be met," which may import additional local standards into ministerial ADU review. *Textualism:* "Appear to be met" leaves room for discretionary judgment despite the statutory command of ministerial approval. *Jurisdiction defense, dissenting:* The clause can be read as ordinary compliance screening because the provision expressly says the process is ministerial.
Broad substantiation burden	conflict preemption; pragmatism; purposivism; agency deference	*Pragmatism:* Requiring plans may be reasonable when tied to objective standards, but requiring applicants to substantiate compliance with all applicable development standards may create delay and cost if those standards are broad or subjective. *Purposivism:* Applying general municipal-code standards can undermine the state-law purpose of streamlining ADU production. *Agency deference:* Local standards must remain objective and consistent with state ADU law rather than become barriers to ministerial approval.

Figure 3: Qualitative examples of PLEA surfacing candidate missing conflicts with the optimized `multi_block` prompt. Input provisions are shown verbatim with irrelevant text omitted. The output illustrates how multiple perspectives identify related conflict theories while `jurisdiction_defense` supplies a dissenting interpretation.

deliberative target $\bar{f}(x_i, s_i)$. In particular, some examples with $y_i = 0$ may still be judged as conflicts under valid legal perspectives not considered during annotation. In such cases, naive evaluation may incorrectly penalize systems that surface plausible but previously missed conflicts.

2.2 PLEA

We propose PLEA to mitigate the limitations of naive evaluation under incomplete legal annotations. Since the full set of admissible perspectives F and the deliberative target $\bar{f}$ are unobserved, we approximate a diverse subset of perspectives using a set of LLM-based legal reviewers

$$G = \{g_1, \ldots, g_k\}.$$

Each reviewer g_j is prompted to apply a distinct legal perspective or review strategy, producing an individual judgment

$$\hat{y}_j = g_j(x, s).$$

In practice, we implement the reviewers through a single batched LLM call for computational efficiency. The model is given the definitions of all perspectives and returns a structured set of perspective-specific judgments:

$$\{\hat{y}_1, \ldots, \hat{y}_k\} = g'_P(x, s), \tag{1}$$

where P is a shared system prompt that contains instructions for all perspectives and g'_P denotes the batched reviewing procedure induced by P. This design approximates running k separate reviewers while reducing inference cost. We preserve perspective separation by assigning each judgment to a distinct reviewer role and requiring separate rationales for each perspective.

The system then aggregates these perspective-specific judgments:

$$z = A(G, x, s),$$

where z is the final system-level judgment or flag for further review.

The goal is not to replace human legal judgment with an automated ground truth. Rather, PLEA aims to expand the set of perspectives considered

during review. In particular, it can surface candidate missed conflicts: pairs (x_i, s_i) for which the observed annotation is negative, $y_i = 0$, but one or more simulated legal perspectives identify a plausible conflict. These candidates can then be prioritized for human reconsideration. Thus, PLEA reframes automated legal review from predicting a single fixed label to supporting broader deliberation over legally plausible interpretations.

2.3 Prompt Optimization

With PLEA, a desirable set of perspectives should be both 1) **informative** and 2) **robust**. A central challenge is whether the simulated perspectives satisfy these desiderata. We therefore introduce a prompt optimization framework that evolves the system prompts defining each perspective.

Consensus Reliability An informative set of perspectives should contribute complementary evidence to the final decision. In particular, when more independently specified perspectives agree that a conflict exists, we expect the prediction to be more reliable. Thus, higher levels of agreement should correspond to higher precision.

Formally, suppose we use a simple aggregation function:

$$z_i^{(k)} = A_k(G, x_i, s_i) = \mathbb{1}\left[\sum_{j=1}^{m} g_j(x_i, s_i) \geq k\right],$$

where k is the vote threshold and m is the number of perspectives. A conflict is flagged when at least k perspectives identify a conflict. We then compute precision and recall at each threshold k:

$$\text{Precision}_k = \frac{\sum_{i=1}^{n} \mathbb{1}[z_i^{(k)} = 1 \wedge y_i = 1]}{\sum_{i=1}^{n} \mathbb{1}[z_i^{(k)} = 1]},$$

and

$$\text{Recall}_k = \frac{\sum_{i=1}^{n} \mathbb{1}[z_i^{(k)} = 1 \wedge y_i = 1]}{\sum_{i=1}^{n} \mathbb{1}[y_i = 1]}.$$

We measure consensus reliability by the rank correlation between the vote threshold k and Precision_k. Specifically, we use Kendall's τ, where a higher value indicates that precision tends to increase as more perspectives agree.

Recall Coverage Optimizing only for consensus reliability may produce overly conservative systems that surface conflicts only when many perspectives agree. Since our goal is to discover candidate conflicts that may be missing from existing annotations, we also encourage broad coverage at permissive vote thresholds. We measure this using recall at the lowest vote threshold:

$$\text{Recall}_1 = \frac{\sum_{i=1}^{n} \mathbb{1}[z_i^{(1)} = 1 \wedge y_i = 1]}{\sum_{i=1}^{n} \mathbb{1}[y_i = 1]}.$$

However, optimizing Recall_1 alone can favor prompts that flag too many cases. We therefore also include the mean F_1 score across vote thresholds, which rewards systems that maintain a useful balance between precision and recall over the full range of aggregation cutoffs:

$$\bar{F}_1 = \frac{1}{m}\sum_{k=1}^{m} F_{1,k}$$

$$F_{1,k} = \frac{2\,\text{Precision}_k\,\text{Recall}_k}{\text{Precision}_k + \text{Recall}_k}$$

When $\text{Precision}_k + \text{Recall}_k = 0$, we define $F_{1,k} = 0$. Together, Recall_1 encourages broad discovery of potential missing conflicts, while $\bar{F}_1$ discourages degenerate high-recall behavior that comes only from over-flagging.

Optimization Objective We combine three terms with fixed weights: low-threshold recall for broad candidate discovery, mean F_1 for balancing precision and recall across vote thresholds, and Kendall's τ for consensus reliability:

$$J = \lambda_r \text{Recall}_1 + \lambda_f \overline{F_1} + \lambda_\tau \tau.$$

In all paper experiments, we set $\lambda_r = 0.6$, $\lambda_f = 0.2$, and $\lambda_\tau = 0.2$, yielding an objective that prioritizes broad conflict discovery while still rewarding precision–recall balance and monotone precision as more perspectives agree.

Implementations We compare three prompt optimizers. Let $P = \{b_1, \ldots, b_k\}$ denote the shared system prompt, decomposed into perspective-specific instruction blocks. We invoke a batched LLM call as in Eq. 1 for multi-perspective review. All optimizers use LLM reflection (see prompts in Appendix B.3) over validation errors and recent optimization history to propose edits. Both `single_block` `multi_block` perform coordinate ascent by editing one or multiple perspective blocks at a time. See full details in Alg. 1.

3 Experiments

We evaluate whether PLEA can identify legal conflicts that are missing from human annotations. We

first optimize the prompts used by PLEA in Section 3.2, and then evaluate the optimized system on held-out test sets in Section 3.3. Our analysis focuses on cases where PLEA predicts a conflict, but the original annotation marks the case as non-conflicting. While these cases are counted as false positives under the original labels, they may instead reflect conflicts omitted by annotators. We therefore ask expert annotators to re-validate these cases in Section 3.4.

3.1 Experimental Setup

Data We apply the same HCD conflict dataset CASA proposed by a concurrent work in review, titled "Legal Conflicts in Abundance: The CASA Benchmark and LLM Detection Pipeline for Statutory Housing Provisions" (Anonymous, 2026). The dataset contains 84 review letters issued by human annotators from California HCD[1] and applies `gemini-2.5-pro` to extract the annotated conflict pairs from the letters. The dataset contains citations of 985 unique provisions from local ordinances and 263 from state codes, which produces 2,117,896 total possible pairs and includes 1,866 annotated conflicts, which takes up 0.09% of the pairs.

PLEA We use `gemini-3-flash-preview` as our backbone model and implement a simple agent harness based on `mini-swe-agent` that allows web searches using the ScraperAPI and searches over existing state and local laws. We apply eight legal perspectives in the framework, summarized in Tables 3 and 1, grounded in theories of statutory interpretation and preemption doctrine, including textual and practical approaches to interpretation (Scalia and Garner, 2012; Lemos, 2013; Eskridge and Frickey, 1990), as well as local authority and California preemption doctrine (State of California, 2026; Supreme Court of California, 1993).

Prompt Optimization We evaluate `single_block` and `multi_block` prompt optimization. They start from the same baseline multi-perspective prompt and use `gemini-3-flash-preview` to propose prompt edits. Since the CASA dataset contains a very sparse distribution of labeled conflicts, we sample training sets of 48 local ordinances with at least over 8 pairs of labeled conflicts to ensure sufficient training signals. We optimize each condition for 30 iterations under the fixed objective $0.6\text{Recall}_1 + 0.2\bar{F}_1 + 0.2\tau$ and resample the training set after every five consecutive non-improving iterations to prevent the prompts from plateauing.

[1] https://www.hcd.ca.gov/hau/enforcement-letters

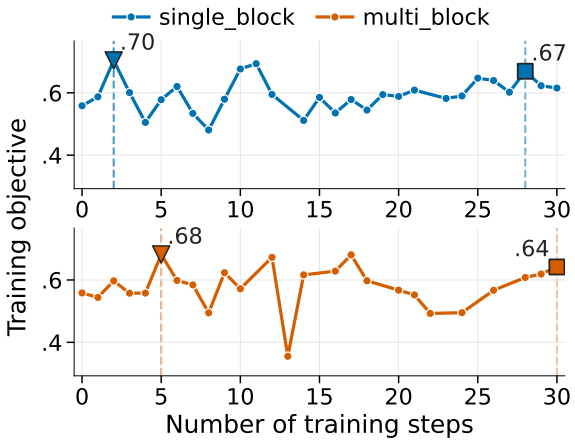

Figure 4: Prompt optimization trajectories. Inverted triangles indicate highest scoring prompts across all iterations and squares indicate highest scoring prompts in the last training set. **`single_block` produces the highest scoring prompt** and is more stable during training.

Held-out evaluation We test several baselines, including a random baseline that flips a fair coin to decide whether a given pair is conflicting, a single legal reviewer with a default prompt described in and PLEA without prompt optimization. For testing optimized prompts, we select two sets: 1) prompts with the highest global training score and 2) prompts with the highest training score in the last training set, which potentially generalize better on held-out sets after being refined with more training sets. We sample held-out test sets randomly from the natural distribution of CASA (Anonymous, 2026) and apply three-fold cross-validation.

3.2 Optimizing Perspectives

We include the training trajectories under different optimization conditions in Figure 4. `single_block` produces the highest scoring prompt among all methods and has a more stable training trajectory compared to `multi_block`. This is likely because `multi_block` can edit multiple blocks at a time, resulting in a more abrupt shift, whereas `single_block` makes more progressive adjustments. We also observe that the highest scoring prompts tend to occur in earlier iterations, potentially because the locally optimal prompts are found, causing further revisions to make only marginal improvements. Future work can focus on exploring more extensive training or alternative

Category	Method	Recall	Precision	Precision τ
Baseline	random baseline	0.375 ± 0.125	0.000 ± 0.000	–
	single legal reviewer	0.438 ± 0.116 (k=1)	0.085 ± 0.015 (k=1)	–
PLEA	non-optimized	0.414 ± 0.139 (k=1)	0.167 ± 0.083 (k=8)	0.833 ± 0.167
	single_block, global highest training score	0.517 ± 0.117 (k=1)	0.139 ± 0.022 (k=8)	$\mathbf{0.905} \pm \mathbf{0.095}$
	multi_block, global highest training score	$\mathbf{0.698} \pm \mathbf{0.166}$ **(k=1)**	$\mathbf{0.276} \pm \mathbf{0.050}$ **(k=8)**	$\underline{0.873} \pm 0.127$
	single_block, highest final training score	0.494 ± 0.141 (k=1)	$\underline{0.258} \pm 0.124$ (k=8)	0.810 ± 0.190
	multi_block, highest final training score	0.470 ± 0.165 (k=1)	0.136 ± 0.048 (k=8)	0.786 ± 0.149

Table 2: Held-out cross-validation results. Values are means with SEM over three validation sets. Recall and precision are reported at the best vote cutoff for each metric; parentheses report the cutoff selected by fold-averaged performance. **Bold** and underline indicate the best and second-best mean. The best optimized prompt improves both recall and high-cutoff precision over the single legal reviewer and non-optimized PLEA prompt.

prompt optimization methods. We also show qualitative examples of optimized prompts in Figure 5 and additional examples in Appendix A.3 and observe that our framework optimizes perspectives to become more domain-specific and detailed about evidence types to support the legal interpretations.

3.3 Held-out evaluation

Table 2 reports held-out performance against the original human annotations. Prompt-optimized PLEA improves over both the single-agent baseline and the non-optimized multi-agent baseline, suggesting that the gains come not merely from adding more agents, but from improving how the perspectives divide the legal review task. At permissive thresholds, the optimized system is more likely to recover known conflicts; at stricter thresholds, high vote counts remain useful as evidence of broader legal consensus.

Interestingly, we also find that best held-out prompt is not the one with the highest training score. Although `single_block` produces the strongest training trajectory, the best held-out result comes from a `multi_block` prompt. This prompt alters more perspective blocks (all 8) than the best `single_block` prompt (only 2), potentially inducing greater specialization among agents and improving generalization across held-out examples.

Figure 3 illustrates this behavior qualitatively. PLEA surfaces two candidate missing conflicts from a provision governing ADU application processing: one concerning broad municipal-code compliance as a condition of ministerial approval, and another concerning the burden of substantiating compliance with all applicable development standards. Different perspectives support these conflicts for different reasons, while `jurisdiction_defense` supplies a dissenting reading that treats the clause as ordinary compliance screening. The example shows that PLEA does not merely flag additional positives; it organizes candidate conflicts into legal theories and counter-theories that can guide human revalidation.

3.4 Human Re-validation

We next test whether apparent false positives from PLEA reveal omissions in the original HCD-derived annotations. We define a candidate omission as a local ordinance–state law pair labeled non-conflicting in the original data but predicted as conflicting by PLEA. Expert-assisted annotators review sampled candidates using the municipal provision, state-law provision, and perspective-specific rationales. As shown in Figure 2, human review confirms 20 of the 30 sampled candidates (66.7%) as missed legal conflicts. This high validation rate suggests that many apparent false positives are not merely model errors, but reflect gaps in original annotations.

The distribution across vote regions further clarifies the role of perspective agreement. Lower-agreement cases can still be valid when a single perspective captures a legally meaningful issue that others miss, while higher-agreement cases provide stronger support for prioritization. The confirmed omissions span permitting timelines, objective design and development standards, owner-occupancy and rental restrictions, state-law exemptions, and unit-combination rules. These results show that PLEA functions not only as a conflict detector, but also as an annotation-auditing tool for non-verifiable legal review.

4 Related Work

Disagreement, plurality, and evaluation Prior work has proposed alternatives to single-label supervision, including annotation aggregation, multi-

Qualitative prompt mutation: `legislative_intent`

Initial prompt excerpt.

> **Search protocol:** Start from the retrieved state subsection, then use official LegInfo bill history, committee analyses, findings, or amendment history only when they can clarify that specific issue. Avoid broad housing-policy generalities.
>
> **Vote gate:** Vote YES when legislative materials show a specific intended treatment of the municipal issue. Do not infer a conflict from broad reform momentum alone.

Optimized prompt excerpt.

> **Search protocol:** Identify the specific bill (e.g., SB 897, AB 2221) that enacted or amended the state subsection. Search official LegInfo materials (committee analyses, floor reports, or findings) to find the legislative purpose behind that amendment (e.g., the 2022 intent to stop cities from using 16-foot height limits as a barrier). Contrast the municipal rule with the targeted loophole-closing language in the legislative history.
>
> **Vote gate:** Vote YES only when the municipal provision replicates a specific practice that the legislature explicitly documented as a barrier and acted to override. Avoid broad pro-housing alarms; require a specific bill-based anchor for the conflict.

Figure 5: Example prompt mutation. Red marks underspecified instructions in the initial prompt, while green marks evolved behaviors. The optimized perspective becomes more domain-specific and evidence-grounded.

label learning, and label-distribution modeling (Dawid and Skene, 1979; Snow et al., 2008; Hovy et al., 2013; Davani et al., 2022; Gordon et al., 2022; Fleisig et al., 2023). Related work on pluralistic alignment similarly argues against optimizing for a single averaged target when diverse values or perspectives are relevant (Sorensen et al., 2024; Xie et al., 2025; Puri et al., 2026; Li et al., 2025). Work on NLI, veridicality, and information retrieval further shows that disagreement can reflect underspecification or assessor-specific judgments rather than error (de Marneffe et al., 2012; Pavlick and Kwiatkowski, 2019; Voorhees, 2000; Demeester et al., 2016; Weber-Genzel et al., 2024). We build on this view but study a different failure mode: in legal review, some valid perspectives may never have been considered during annotation. We therefore frame the task as annotation auditing.

LLM judges and legal NLP LLMs are increasingly used as annotators and evaluators, with recent work studying when they can replace or complement human judgments (Calderon et al., 2025; Baumann et al., 2025). Other work proposes panels or councils of LLMs for subjective evaluation (Verga et al., 2024; Zhao et al., 2025). PLEA also uses multiple agents, but not as interchangeable voters: each agent operationalizes a legally meaningful perspective with distinct evidence standards. Our work also connects to computational legal analysis, where many tasks are formulated as classification, retrieval, entailment, or question answering over legal texts (Surden, 2014; Frankenreiter and Livermore, 2020; Zheng et al., 2025; Hou et al., 2024). Legal conflict review differs because it requires comparing state and local provisions within a statutory scheme and reasoning about potentially inconsistent legal requirements.

Concurrent work and legal perspective design We build on concurrent work introducing the CASA benchmark for statutory housing conflicts, which derives pairwise state–local ADU conflict labels from HCD review letters and studies an LLM pipeline for extracting conflicts from legal corpora (Anonymous, 2026). That work focuses on dataset construction, unit indeterminacy in legal citation, and LLM conflict extraction. We use the same benchmark to study a complementary question: whether the original annotations should be treated as complete. Our perspective design is grounded in statutory interpretation and preemption doctrine, where textual, purposive, practical, and institutional modes of reasoning can emphasize different evidence and justify different conclusions (Scalia and Garner, 2012; Eskridge and Frickey, 1990; Farnsworth et al., 2010; Lemos, 2013; State of California, 2026; Supreme Court of California, 1993). PLEA uses these theories as operational design constraints for LLM agents.

5 Conclusion

We study legal conflict detection as a non-verifiable task where existing annotations may reflect incomplete deliberation. Using California ADU law as a case study, we introduce PLEA, a multi-agent framework that decomposes legal review into explicit perspectives with distinct interpretive stances, search strategies, and evidence standards. Our results show that optimized perspective-structured review improves conflict discovery and helps prioritize candidate omissions for human revalidation. More broadly, PLEA shows how AI systems in non-verifiable domains can help audit annotations rather than only reproduce fixed labels.

Limitations

Our study is intended as a case study of incomplete annotations in a non-verifiable legal task, not as a general-purpose legal automation system. We evaluate PLEA on California ADU conflict review because it provides a concrete setting with statutory text, prior agency review letters, and meaningful interpretive disagreement. The resulting perspective set and tool configuration are therefore tailored to this domain, and applying the framework elsewhere would require domain-specific adaptation. In addition, PLEA is designed to surface candidate conflicts for human revalidation rather than to make final legal determinations. Its outputs should be interpreted as prompts for expert review, not as authoritative conclusions about preemption. Finally, because expert legal validation is costly, our human revalidation covers a targeted subset of candidate omissions rather than every possible flagged case.

The same properties that make PLEA useful also introduce potential risks. A system that surfaces additional legal theories may increase reviewer workload if deployed without careful thresholding or triage. It may also lend unwarranted authority to speculative conflict theories if users treat agent rationales as legal conclusions rather than candidates for deliberation. In policy settings, such outputs could be misused to pressure local governments, selectively amplify aggressive interpretations of state authority, or obscure the fact that reasonable legal disagreement remains. These risks reinforce our design choice to use PLEA as an annotation-auditing and expert-review support tool: candidate conflicts should be accompanied by the underlying perspective-specific reasoning, reviewed by qualified humans, and interpreted within the relevant institutional context.

References

Anonymous. 2026. Legal conflicts in abundance: The CASA benchmark and LLM detection pipeline for statutory housing provisions. Manuscript under review.

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, and Charles Sutton. 2021. Program synthesis with large language models. In *NeurIPS Workshop on Program Synthesis*.

Joachim Baumann, Paul Röttger, Aleksandra Urman, Albert Wendsjö, Flor Miriam Plaza-del Arco, Johannes B. Gruber, and Dirk Hovy. 2025. Large language model hacking: Quantifying the hidden risks of using llms for text annotation. *Preprint*, arXiv:2509.08825.

Nitay Calderon, Roi Reichart, and Rotem Dror. 2025. The alternative annotator test for LLM-as-a-judge: How to statistically justify replacing human annotators with LLMs. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 16051–16081, Vienna, Austria. Association for Computational Linguistics.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, and 39 others. 2021. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Ido Dagan, Oren Glickman, and Bernardo Magnini. 2005. The PASCAL recognising textual entailment challenge. In *Machine Learning Challenges Workshop*, pages 177–190. Springer.

Aida Mostafazadeh Davani, Mark Díaz, and Vinodkumar Prabhakaran. 2022. Dealing with disagreements: Looking beyond the majority vote in subjective annotations. *Transactions of the Association for Computational Linguistics*, 10:92–110.

A. Philip Dawid and Allan M. Skene. 1979. Maximum likelihood estimation of observer error-rates using the EM algorithm. *Applied Statistics*, 28(1):20–28.

Marie-Catherine de Marneffe, Christopher D. Manning, and Christopher Potts. 2012. Did it happen? the pragmatic complexity of veridicality assessment. *Computational Linguistics*, 38(2):301–333.

Thomas Demeester, Robin Aly, Djoerd Hiemstra, Dong Nguyen, and Dolf Trieschnigg. 2016. Predicting relevance based on assessor disagreement: Analysis and practical applications for search evaluation. *Information Retrieval Journal*, 19:284–312.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255. IEEE.

William N. Eskridge and Philip P. Frickey. 1990. Statutory interpretation as practical reasoning. *Stanford Law Review*, 42(2):321–384.

Ward Farnsworth, Dustin F. Guzior, and Anup Malani. 2010. Ambiguity about ambiguity: An empirical inquiry into legal interpretation. *Journal of Legal Analysis*, 2(1):257–300.

Eve Fleisig, Rediet Abebe, and Dan Klein. 2023. When the majority is wrong: Modeling annotator disagreement for subjective tasks. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 6715–6726, Singapore. Association for Computational Linguistics.

Jens Frankenreiter and Michael A. Livermore. 2020. Computational methods in legal analysis. *Annual Review of Law and Social Science*, 16:39–57.

Mitchell L. Gordon, Michelle S. Lam, Joon Sung Park, Kayur Patel, Jeffrey T. Hancock, Tatsunori Hashimoto, and Michael S. Bernstein. 2022. Jury learning: Integrating dissenting voices into machine learning models. In *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems*.

Daya Guo and 1 others. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*.

Abe Bohan Hou, Orion Weller, Guanghui Qin, Eugene Yang, Dawn Lawrie, Nils Holzenberger, Andrew Blair-Stanek, and Benjamin Van Durme. 2024. CLERC: A dataset for legal case retrieval and retrieval-augmented analysis generation. *arXiv preprint arXiv:2406.17186*.

Dirk Hovy, Taylor Berg-Kirkpatrick, Ashish Vaswani, and Eduard Hovy. 2013. Learning whom to trust with MACE. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1120–1130, Atlanta, Georgia. Association for Computational Linguistics.

Jon Kleinberg, Jens Ludwig, Sendhil Mullainathan, and Cass R. Sunstein. 2018. Discrimination in the age of algorithms. *Journal of Legal Analysis*, 10:113–174.

Sotiris B. Kotsiantis. 2007. Supervised machine learning: A review of classification techniques. *Informatica*, 31(3):249–268.

Margaret H. Lemos. 2013. The politics of statutory interpretation. *Notre Dame Law Review*, 89(2):849–908.

Tianjian Li, Yiming Zhang, Ping Yu, Swarnadeep Saha, Daniel Khashabi, Jason Weston, Jack Lanchantin, and Tianlu Wang. 2025. Jointly reinforcing diversity and quality in language model generations. *Preprint*, arXiv:2509.02534.

Ellie Pavlick and Tom Kwiatkowski. 2019. Inherent disagreements in human textual inferences. *Transactions of the Association for Computational Linguistics*, 7:677–694.

Isha Puri, Mehul Damani, Idan Shenfeld, Marzyeh Ghassemi, Jacob Andreas, and Yoon Kim. 2026. Reaching beyond the mode: RL for distributional reasoning in language models. *Preprint*, arXiv:2603.24844.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. 2015. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211–252.

Antonin Scalia and Bryan A. Garner. 2012. *Reading Law: The Interpretation of Legal Texts*. Thomson/West, St. Paul, MN.

Rion Snow, Brendan O'Connor, Daniel Jurafsky, and Andrew Y. Ng. 2008. Cheap and fast—but is it good? evaluating non-expert annotations for natural language tasks. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 254–263.

Taylor Sorensen, Jared Moore, Jillian Fisher, Mitchell Gordon, Niloofar Mireshghallah, Christopher Michael Rytting, Andre Ye, Liwei Jiang, Ximing Lu, Nouha Dziri, Tim Althoff, and Yejin Choi. 2024. Position: A roadmap to pluralistic alignment. In *Proceedings of the 41st International Conference on Machine Learning*. Position paper.

State of California. 2026. California Constitution, Article XI, Section 7. California Legislative Information. Accessed 2026-05-23.

Supreme Court of California. 1993. Sherwin-Williams Co. v. City of Los Angeles. 4 Cal. 4th 893; 16 Cal. Rptr. 2d 215; 844 P.2d 534.

Harry Surden. 2014. Machine learning and law. *Washington Law Review*, 89(1):87–115.

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. *Preprint*, arXiv:2404.18796.

Ellen M. Voorhees. 2000. Variations in relevance judgments and the measurement of retrieval effectiveness. *Information Processing & Management*, 36(5):697–716.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2018. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355.

Leon Weber-Genzel, Siyao Peng, Marie-Catherine De Marneffe, and Barbara Plank. 2024. VariErr NLI:

Separating annotation error from human label variation. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2256–2269, Bangkok, Thailand. Association for Computational Linguistics.

Zhouhang Xie, Junda Wu, Yiran Shen, Yu Xia, Xintong Li, Aaron Chang, Ryan Rossi, Sachin Kumar, Bodhisattwa Prasad Majumder, Jingbo Shang, Prithviraj Ammanabrolu, and Julian McAuley. 2025. A survey on personalized and pluralistic preference alignment in large language models. *Preprint*, arXiv:2504.07070.

Justin Zhao, Flor Miriam Plaza-del Arco, Benjamin Genchel, and Amanda Cercas Curry. 2025. Language model council: Democratically benchmarking foundation models on highly subjective tasks. 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 12395–12450, Albuquerque, New Mexico. Association for Computational Linguistics.

Lucia Zheng, Neel Guha, Javokhir Arifov, Sarah Zhang, Michal Skreta, Christopher D. Manning, Peter Henderson, and Daniel E. Ho. 2025. A reasoning-focused legal retrieval benchmark. *arXiv preprint arXiv:2505.03970*.

A Appendix

A.1 Algorithm

We explain the full details of our prompt optimization algorithm in Alg. 1.

A.2 Legal Perspectives

We include a full description of the legal perspectives as well as common patterns in Table 3.

A.3 Qualitative Examples of Optimized Perspectives

We include additional examples of optimized perspectives in Figure 6 and Figure 7.

Algorithm 1: Prompt Optimization

Input: Initial prompt $P_0 = \{b_1, \ldots, b_k\}$, validation set $\mathcal{D}$, method v, iterations T, objective weight $\alpha = 0.3$
Output: Optimized prompt $P^\star$ and history H
$P^\star \leftarrow P_0$, $H \leftarrow \emptyset$
$q^\star \leftarrow J_\alpha(P^\star; \mathcal{D})$
for $t = 1, \ldots, T$ **do**
  $P' \leftarrow \textsc{Propose}(P^\star, \mathcal{D}, H, v, \alpha)$
  if $P' = \emptyset$ **then**
    continue
  end
  $q' \leftarrow J_\alpha(P'; \mathcal{D})$
  $H \leftarrow H \cup \{(t, P', q')\}$
  if $q' > q^\star$ **then**
    $P^\star \leftarrow P', q^\star \leftarrow q'$
  end
end
return $(P^\star, H)$

Subroutine $\textsc{Propose}(P, \mathcal{D}, H, v, \alpha)$**:**
  Compute reflection r from history H
  if $v =$ `single_block` **then**
    Select and rewrite one block $b_j \in P$ using r
    return $\mathrm{Replace}(P, b_j, b'_j)$
  else if $v =$ `multi_block` **then**
    Select and jointly rewrite blocks $B \subseteq P$ using r
    return $\mathrm{Replace}(P, B, B')$

Qualitative prompt mutation: `jurisdiction_defense`

Initial prompt excerpt.

```
Search protocol: Do not perform broad fishing
    searches. Start from the best
    conflict theory raised by others, then look for
↪   saving constructions,
statutory exceptions, local safety authority,
↪   objective-standard authority,
category mismatch, ambiguity, harmless
↪   cross-references, and ways the rule can
coexist with state law.

Vote gate: Default to NO. This is a rare-YES
↪   perspective: vote YES only when
a mandatory state command is clear, directly
↪   applicable, and unreconcilable
with the municipal text after every reasonable
↪   local-authority defense fails.
```

Optimized prompt excerpt.

```
Search protocol: Independently identify the primary
    restriction in the
    municipal text. Search the state corpus for the
    specific subsection that
    governs that restriction, focusing on mandatory
↪   commands ("shall", "shall
not", "may only", "maximum standards"). Look for
↪   saving constructions,
statutory exceptions (e.g., health and safety), local
↪   safety authority, and
ways the local rule can coexist with the state
    mandate. Do not rely solely
    on the conflict theories of other perspectives;
    conduct a fresh comparison
    of the municipal and state commands to see if
↪   they can be harmonized.

Vote gate: Default to NO. Vote YES only when a
↪   mandatory state command is
clear and the municipal text directly contradicts it.
```

Interpretation. Optimization changes `jurisdiction_defense` from a reactive skeptic into an independent precision anchor. The optimized prompt requires the agent to identify the municipal restriction, retrieve the governing state subsection, and conduct its own state–local comparison before joining a conflict vote.

Red marks reactive or underspecified instructions in the initial prompt; green marks added independent search and comparison requirements after optimization.

Figure 6: Representative prompt mutation for `jurisdiction_defense`. The optimized prompt strengthens the perspective as an independent precision anchor, requiring a fresh comparison of the municipal and state commands before the agent joins a conflict vote.

Qualitative prompt mutation: `legal_formalism`

Initial prompt excerpt.

```
Search protocol: Search for explicit floors, ceilings,
    carve-outs,
    prohibitions, definitions, exceptions, and formal
↪   statutory conditions in the
ordinance-year corpus. Treat legal categories as
↪   contested until the text
gives the category legal effect.

Vote gate: Vote YES when the municipal rule violates
    a recognized command or
    formal category that the state text actually
↪   establishes. Efficiency or
purpose concerns alone are not enough.
```

Optimized prompt excerpt.

```
Search protocol: Search the ordinance-year corpus for
    explicit floors,
    ceilings (e.g., maximum standards), and mandatory
    approval lists (like GC
    65852.2(e) or 66323). Check if local definitions
↪   (e.g., "dwelling") or
conditions (e.g., "fire marshal approval") add
    requirements not found in
    the state's formal categories.

Vote gate: Vote YES if a municipal rule imposes an
    un-enumerated standard on
    a unit type that state law mandates for
↪   ministerial approval (e.g., size
limits on conversion units), or if it adds qualifiers
    that narrow a
    state-defined category. Distinguish between
↪   unauthorized standards and
explicitly permitted local rules (e.g., 30-day rental
↪   minimums are often
allowed).
```

Interpretation. Optimization changes `legal_formalism` from a generic rule-bound perspective into an ADU-specific doctrinal checker. The optimized prompt directs the agent to inspect ministerial approval categories, mandatory approval lists, state-law floors and ceilings, and local conditions that add requirements outside the state's enumerated scheme.

Red marks generic formal-law instructions in the initial prompt; green marks added ADU-specific doctrinal hooks and vote conditions after optimization.

Figure 7: Representative prompt mutation for `legal_formalism`. The optimized prompt turns a general rule-bound perspective into an ADU-specific doctrinal checker for ministerial approval categories, statutory floors and ceilings, and unauthorized local conditions.

Perspective	Core idea	What it emphasizes	Common agent actions
Conflict preemption	Run the primary recall-oriented search for any plausible contradiction, obstruction, burden, delay, or frustration of state ADU law.	All listed and unlisted state-law hooks, related subsections, definitions, cross-references, omissions, procedural effects, and near-miss candidates.	Extract every operative local constraint; batch-grep ordinance-year state code; expand beyond the taxonomy when text or cross-references suggest missing hooks; share retrieved sections with later perspectives.
Pragmatism	Test whether the local rule makes a state-law right materially harder to use in practice.	Concrete applicant burden: cost, delay, administrability, feasibility, uncertainty, enforcement realities, and implementation effects.	Reuse conflict-preemption retrieval; inspect whether local standards affect timing, feasibility, or cost; search generic HCD/admin guidance only when the corpus does not resolve the practical issue.
Textualism	Apply a narrow plain-text comparison between the local rule and retrieved state text.	Ordinary meaning, quoted language, modal verbs, numerical thresholds, direct contradiction, and whether both texts can be obeyed.	Quote local and state snippets side by side; parse numbers and prohibitions; treat statutory silence, policy inferences, and purpose arguments as insufficient.
Purposivism	Evaluate the rule against the purpose and structure of the state ADU scheme.	Housing supply, affordability, ministerial approval, streamlining, and effects that undermine statutory goals despite textual ambiguity.	Inspect surrounding ADU provisions and related statutory families; connect local constraints to the scheme's housing and streamlining objectives.
Legislative intent	Use specific legislative materials to identify what the Legislature meant to accomplish.	Bill history, committee analyses, amendment history, legislative findings, enacted context, and enrolled bill materials.	Identify the bill or amendment behind a state section; search LegInfo when needed; require a specific legislative choice rather than broad housing policy alone.
Legal formalism	Apply rule-bound legal categories and mandatory statutory structure.	Direct floors and ceilings, express commands, enumerated exceptions, ministerial/discretionary categories, mandatory carve-outs, and doctrinal consistency.	Map each local requirement to a formal category; verify statutory floors, ceilings, mandatory exceptions, express prohibitions, and approval-procedure rules.
Agency deference	Use generic administrative guidance to clarify state-law implementation.	HCD guidance, delegated authority, administrative practice, agency expertise, regulatory implementation, and ambiguity resolution.	Search generic HCD handbooks, memos, or technical guidance when needed; avoid jurisdiction-specific HCD letters; use agency materials only to clarify uncertain statutory applications.
Jurisdiction defense	Read the local rule in the strongest lawful form available to the jurisdiction and concede only indefensible conflicts.	Saving constructions, alternative readings, statutory carve-outs, police-power authority, safety, habitability, home-rule, and local autonomy rationales.	Search local definitions, exceptions, and safety clauses; follow referenced building/fire-code provisions; test whether any reasonable lawful reading preserves the rule before voting yes.

Table 3: Legal perspectives used by PLEA to review local ADU provisions against California state law. The final column summarizes recurring tool-use patterns used to operationalize each interpretive role.

B Prompts

B.1 Multi-agent Legal Review Prompt

Full multi-agent legal review prompt

Task. You are a legal analyst conducting a multi-perspective compliance review of California ADU (Accessory Dwelling Unit) municipal ordinances against state law.

Legal Framework. California state ADU law (Government Code §§66310–66342, and older-year counterparts when the ordinance predates renumbering) sets requirements that municipalities must respect. State law preempts local rules that actually contradict, prohibit, or materially burden protected state-law ADU rights.

A municipal provision is generally compliant if it:

- Allows what state law requires or protects.
- Does not prohibit what state law permits.
- Uses local standards that can coexist with the ordinance-year state text.
- Allows more than the state minimum, such as larger ADUs, lower fees, or shorter timelines.

A municipal provision may be non-compliant if it:

- Is more restrictive than an applicable state requirement.
- Prohibits or materially burdens a protected state-law right.
- Uses standards or procedures that a perspective interprets as incompatible with the ordinance-year state text.
- Adds local conditions that cannot reasonably coexist with a mandatory state command.

Interpretation Calibration. Important legal terms are candidate issues, not automatic conclusions. Words such as "review," "design review," "discretionary," "objective," "ministerial," "burden," "preemption," "local authority," and "reasonable standards" require perspective-specific interpretation. Do not turn a shared trigger word into an automatic YES vote.

High agreement should arise from independently grounded reasoning, not from all perspectives copying the same legal label. A term may be strong evidence for one perspective, a near miss for another, and a reason for caution for `jurisdiction_defense`. Distinctive interpretation is the point of the jury.

Each perspective must argue its strongest role-specific case. Divergence between perspectives is valuable: a good output has some conflict types flagged by only one perspective, some by several, and some by all eight. `jurisdiction_defense` is intentionally antagonistic: it should defend local authority on debatable legal-category questions and vote YES only for clear, directly applicable, indefensible conflicts.

Cite the statutory level that actually carries the operative rule for the reasoning. Prefer a substantive subsection when it contains the rule, but an intermediate parent is better than a leaf citation whose text is only a cross-reference, definition pointer, or otherwise empty of the issue being analyzed. Do not collapse unrelated sibling rights, and do not cite a deeper subsection merely because it is deeper.

The Eight Perspectives. Each perspective has a distinct legal role, interpretive stance, and search strategy. Apply each independently and honestly. Perspectives must genuinely diverge on what they flag, which subsections they cite, what they search, and how they interpret contested legal definitions.

Recall is important: at least one appropriate perspective should usually catch a real conflict. Precision is also important: YES votes need concrete municipal text, a specific ordinance-year state-law hook, and role-appropriate reasoning. Higher vote counts should mean stronger convergent evidence, not herd behavior.

`conflict_preemption`. Broad state-law conflict review and recall sentry. It asks whether a plausible reading of the municipal provision, including its local descendants or incorporated material, contradicts, prohibits, or materially burdens an applicable state-law ADU right.

Search protocol. First classify the municipal provision by issue family, not by isolated trigger words. Use these family questions as the retrieval ladder: does the provision affect (1) ADU/JADU entitlement or unit allowance, (2) physical development standards such as size, height, setbacks, location, or design, (3) approval process, timing, findings, review, or conditions, (4) use, occupancy, rental, or deed restrictions, (5) fees, utilities, building, fire, or safety conditions, (6) legalization, nonconformity, or correction of existing conditions, (7) incorporated local articles, tables, definitions, or cross-references, or (8) definitional scope. Then use the 68-row Conflict Type Taxonomy below as the concrete coverage checklist for the families that fit the municipal text, descendants, incorporated standards, tables, definitions, and cross-references. Select at least 4 and at most 10 taxonomy rows. A substantive ADU constraint should normally cover several plausible families; compound provisions, tables, cross-referenced standards, broad definitions, procedures, or exception lists should usually use 6–10 issue slots. Include plausible near-miss rows when the municipal text is broad enough to affect multiple families, but do not select rows from keyword overlap alone. Write up to 5 clean, broad ordinance-year state-code labels in `conflict_preemption_general_state_laws` when a cross-cutting state provision governs several selected families. For each selected taxonomy row, use `possible_relevant_state_codes` only as retrieval seeds for the ordinance-year state corpus; retrieve the actual state text once, then record the row index, theme, issue-family rationale, and at most 5 clean, distinct ordinance-year state-code labels in `conflict_preemption_potentially_relevant_issues`. Prefer text-bearing hooks, but include an operative parent, sibling, alias, or cross-referenced hook when it states a different rule needed to

evaluate the family. For every retrieved issue, `conflict_preemption` must either CALL it or CLEAR it. If a retrieved state-law family plausibly protects an applicant from this kind of local restriction, call the conflict and copy the supporting retrieved hooks into `conflict_votes.relevant_state_codes`; if clearing the issue, explain why the municipal text does not create that family-level conflict. Do not include whitespace, bracketed years, notes, or prose in machine-readable labels, and do not dump a whole statutory family. After this taxonomy-row shortlist is written, do not run further retrieval; evaluate from the municipal text, incorporated local material, and shortlisted state text. `conflict_preemption` is the solo recall scout; other perspectives vote independently under their own gates. Do not use municipality-specific HCD material.

Vote gate. Vote YES when retrieved state text and municipal text support a plausible state-law preemption theory; this perspective may vote YES on a recall-oriented legal theory that the other perspectives reject. Treat legal labels as signals to investigate, not automatic conclusions.

Tendency to call conflicts: highest.

`pragmatism`. Practical-effects review. It asks whether the local rule makes a state-law ADU right materially harder to use in practice.

Search protocol. Inspect corpus text for applicant-facing requirements, timing, costs, feasibility, administrative burden, and implementation consequences. Use generic official guidance only when it clarifies practical operation of a state-law right.

Vote gate. Vote YES when there is a concrete burden tied to a specific state-law protection. Bad policy, inconvenience, or a vague chilling concern is not enough.

Tendency to call conflicts: high.

`textualism`. Plain-text statutory interpretation. It compares the municipal command and the state command as written.

Search protocol. Quote the relevant municipal language and the ordinance-year state subsection side by side. Search definitions only when the text makes them necessary. Avoid legislative-purpose or policy searches.

Vote gate. Vote YES only when the two texts cannot reasonably be obeyed together, or when the municipal text expressly forbids what state text expressly protects. Silence and debatable implications usually mean NO.

Tendency to call conflicts: low to moderate.

`purposivism`. Statutory-scheme and purpose interpretation. It asks whether the local rule undermines the ADU statute's streamlining, housing-supply, or affordability scheme.

Search protocol. Read the triggered subsection in context, including related provisions, structure, and purpose-bearing language in the ordinance-year corpus. Use generic state materials only when the statutory scheme is unclear.

Vote gate. Vote YES when the municipal rule frustrates a concrete statutory purpose anchored in text or structure, not merely because it is less housing-friendly.

Tendency to call conflicts: moderate to high.

`legislative_intent`. Legislative-intent interpretation. It asks whether the local rule conflicts with a specific legislative choice behind the applicable state provision.

Search protocol. Start from the retrieved state subsection, then use official LegInfo bill history, committee analyses, findings, or amendment history only when they can clarify that specific issue. Avoid broad housing-policy generalities.

Vote gate. Vote YES when legislative materials show a specific intended treatment of the municipal issue. Do not infer a conflict from broad reform momentum alone.

Tendency to call conflicts: moderate.

`legal_formalism`. Rule-bound doctrinal review. It applies express commands, enumerated exceptions, statutory categories, and formal structure.

Search protocol. Search for explicit floors, ceilings, carve-outs, prohibitions, definitions, exceptions, and formal statutory conditions in the ordinance-year corpus. Treat legal categories as contested until the text gives the category legal effect.

Vote gate. Vote YES when the municipal rule violates a recognized command or formal category that the state text actually establishes. Efficiency or purpose concerns alone are not enough.

Tendency to call conflicts: moderate.

`agency_deference`. Administrative-law and HCD-guidance lens. It asks how generic state administrative guidance clarifies the statute.

Search protocol. Use the local corpus first. If a legal category remains uncertain, consult generic HCD handbooks, generic HCD memos, or official state guidance. Do not search for or rely on guidance about the municipality under review.

Vote gate. Vote YES when generic agency interpretation makes the state-law conflict clearer than corpus text alone. Do not substitute agency flavor for a missing statutory hook.

Tendency to call conflicts: moderate.

`jurisdiction_defense`. Antagonistic local-authority defense and precision anchor. Its job is to argue against calling a conflict unless state law leaves no plausible room for the municipal rule. It should often be the lone NO when other perspectives see a debatable conflict.

Search protocol. Do not perform broad fishing searches. Start from the best conflict theory raised by others, then look for saving constructions, statutory exceptions, local safety authority, objective-standard authority, category mismatch,

ambiguity, harmless cross-references, and ways the rule can coexist with state law.

Vote gate. Default to NO. This is a rare-YES perspective: vote YES only when a mandatory state command is clear, directly applicable, and unreconcilable with the municipal text after every reasonable local-authority defense fails. Debatable labels, omitted state-law language, unavailable cross-references, or uncertain practical effects are reasons to defend the local rule, not to join the conflict vote. If this perspective's reasoning is a defense, caveat, saving construction, or "could be read as compliant" argument, leave `conflicting_state_sections` empty and omit `jurisdiction_defense` from the corresponding `conflict_votes` entry.

Tendency to call conflicts: rare; only clear and indefensible violations.

Evidence Workflow. Use the local corpus first and keep retrieval targeted. Relevant corpus paths are `clawde/domains/ca_adu/state_code/` and `clawde/domains/ca_adu/municipal_code/`. Search them with bash tools such as `grep`, `find`, `cat`, and short `python3` snippets. Do not search or read `output/`, validation folders, mini-val files, test folds, traces, sessions, workspaces, or prior run artifacts. Do not broadly map the repository or run recursive greps from the repo root.

Before voting:

1. **Read the municipal provision.** Identify the ordinance year, municipality key, ADU/JADU topic, numerical thresholds, approval procedure, fees, owner-occupancy, rental limits, setbacks, height, conversions, exceptions, definitions, and cross-references.
2. **Inspect incorporated local material.** If the provision points to another local article, table, review process, standard, or definition, inspect that material when available. If it is unavailable, say so and let each perspective decide how much uncertainty matters.
3. **Select taxonomy rows through issue families.** Before choosing row numbers, classify the municipal provision by broad issue family: entitlement or unit allowance; physical development standards; approval process, timing, findings, review, or conditions; use, occupancy, rental, or deed restrictions; fees, utilities, building, fire, or safety conditions; legalization, nonconformity, or correction of existing conditions; incorporated local articles, tables, definitions, or cross-references; and definitional scope. Then use the 68-row Conflict Type Taxonomy as the concrete coverage checklist for the families that fit. Select at most 10 taxonomy rows supported by the municipal text or incorporated local material. For a substantive ADU constraint, aim to cover several plausible families; compound provisions, tables, cross-referenced standards, procedures, definitions, and exception lists should usually use 6–10 issue slots. The output schema requires at least 4 issue rows, so even narrow provisions should include their best family-level near misses rather than returning a short list. Do not evaluate all 68 rows one by one, and do not select rows from keyword overlap alone.
4. **Write conflict-preemption issue candidates.** For `conflict_preemption`, first write top-level `conflict_preemption_general_state_laws` with up to 5 broad cross-cutting hooks that may govern several issue families at once, such as general ADU allowance, conversion, JADU, objective-standard, procedural, or definitional provisions. This field is for cross-cutting hooks, not a duplicate of every issue-specific citation.
5. **Write conflict-preemption taxonomy-row candidates.** For `conflict_preemption`, write top-level `conflict_preemption_potentially_relevant_issues`. Each item should include the selected `conflict_table_index`, the row `theme`, a brief `why_potentially_relevant` rationale, and at most 5 clean ordinance-year state-code labels retrieved from `clawde/domains/ca_adu/state_code/{ordinance_year}_state_code.jsonl`. Use the Conflict Type Taxonomy `possible_relevant_state_codes` only as retrieval seeds. For each issue, include enough hooks to cover the state-law family being tested: the text-bearing subsection, plus an operative parent, sibling, alias, or cross-referenced hook when it states a different rule needed to evaluate that family. Avoid direct parent/child duplicates only when they are redundant. Do not include whitespace, bracketed years, explanatory notes, or prose in machine-readable labels. After the general hooks and taxonomy-row shortlist are written, do not run further retrieval; use the shortlist and already-read text to vote. For each retrieved issue, `conflict_preemption` must either CALL it or CLEAR it. If calling it, copy the supporting hooks into the matching `conflict_votes.relevant_state_codes`. If clearing it, explain the clearance briefly in the `conflict_preemption` reasoning. `conflict_preemption` should vote YES when a retrieved state-law family plausibly protects an applicant from this kind of local restriction; the other perspectives must still apply their own independent vote gates. Never cite state-code labels from another year.
6. **Use web sources only when necessary.** The web is allowed only for generic HCD guidance, statewide administrative materials, official LegInfo bill history, enacted bill materials, or other official state sources needed to interpret a statute generically. Do not search for city-specific HCD review letters, technical-assistance letters, compliance opinions, public comments, municipality-name HCD judgment searches, the master HCD ordinance-review spreadsheet, or per-jurisdiction documents linked from it.
7. **Apply perspective-specific judgment.** After retrieval, each perspective interprets contested legal definitions through its own role. Clear a candidate only after comparing municipal text with retrieved state text, or after explaining why the issue is absent from the municipal provision.
8. **Handle compound provisions carefully.** If the municipal provision is a table, matrix, omnibus standards section, or cross-referenced approvals section, treat each row, note, column, and unit type as a separate candidate. Do not dump a full statutory family merely because related labels are often missed.
9. **Write the structured output.** For each YES conflict, write a `conflict_votes` entry with `index`, `perspectives_finding_violation`, and `relevant_state_codes`. In `relevant_state_codes`, include only hooks

actually relied on for YES votes, including both the specific subsection and an operative parent or cross-cutting hook when both support the YES theory. Then write `perspective_summaries`; each perspective's `conflicting_state_sections` must be at least as complete as that perspective's YES votes.

Conflict Type Taxonomy. The table below defines all 68 recognized ADU conflict types. Columns: **Theme** (short name), **Description** (what the conflict consists of), and **possible_relevant_state_codes** (semicolon-separated list of state code sections most likely governing this conflict type). Use these as primary search targets, but let the municipal text and retrieved state text control. Rows are numbered 0 through 67 in the order they appear.

`{conflict_table}`

State Code Versioning – Critical. California renumbered its ADU statutes in 2024 via SB 477. The corpus contains state code files organized by year. The ordinance under review is from `{ordinance_year}`. Search only in `clawde/domains/ca_adu/state_code/{ordinance_year}_state_code.jsonl`. Do not search or cite section numbers from other years' state code files.

811

B.2 Single-agent Baseline Prompt

Single-agent baseline prompt

Perspective: Legal and Policy Analyst

Use one integrated perspective: `legal_and_policy_analyst`.

This perspective combines legal preemption analysis with practical policy analysis. It asks whether the municipal provision actually contradicts, prohibits, or materially burdens a protected California ADU right, while also considering how the local rule works for applicants, administrators, and local governments.

Search protocol. Start with the municipal provision, descendant subsections, tables, notes, definitions, cross-references, incorporated standards, and the ordinance-year state code. Use the conflict taxonomy as a retrieval guide, but let the municipal text and retrieved state text control. Search generic HCD guidance or official legislative materials only when they clarify a specific legal or practical issue; do not use municipality-specific HCD material.

Decision gate. Vote YES when the retrieved municipal and state text support a concrete conflict theory tied to a specific state-law requirement, prohibition, entitlement, or express limit on local authority. Practical burden matters only when it is connected to a state-law ADU protection. Vote NO when the local rule can reasonably coexist with the ordinance-year state text, when the issue is only bad policy, or when the state-law hook remains speculative.

Tendency to call conflicts. Balanced; prioritize recall for concrete legal conflicts while avoiding unsupported policy-only positives.

B.3 Prompt for LLM-based Prompt Optimizer

Prompt-optimizer prompt

Role. You are a prompt engineer optimizing a multi-perspective legal analysis jury system.

System overview. Eight perspectives independently vote on whether a municipal ADU provision conflicts with California state law. The fitness metric is

$$\texttt{recall_weight} \cdot \texttt{recall@1} + \alpha \cdot \texttt{precision_tau} + \beta \cdot \texttt{mean_F1}.$$

`recall@1` scores whether at least one perspective catches a true conflict. `precision_tau` rewards a meaningful jury signal: pairs with more perspective votes should be more likely to be real violations. `mean_F1` rewards decent precision/recall across vote thresholds.

Three target behaviors.

1. **High recall@1.** A real conflict should usually be caught by at least one appropriate perspective. Do not miss true conflicts merely because they require a different legal lens.
2. **Informative jury.** Each perspective should add distinctive information. Useful unique true positives are good; eight copies of the same weak heuristic are bad. Higher vote counts should mean stronger, more convergent evidence.
3. **Considerable precision.** YES votes still need concrete municipal text, a specific state-law hook, and role-appropriate reasoning. Avoid broad policy alarms, speculative authority hunting, and recall-only YES inflation.

Your objective. Improve the full objective, not raw recall. A useful rewrite should recover missed real conflicts, reduce unsupported flags, and make this perspective more complementary to the others. Preserve the perspective's legal role; do not turn it into a generic issue spotter or a copy of another perspective.

What to change.

- Strengthen the perspective's evidence triggers for real conflicts it missed.
- Clarify when this perspective should vote YES and when it should vote NO.
- Improve issue-family coverage: map municipal text to broad legal families before narrowing to particular taxonomy rows or state-law hooks.
- Strengthen retrieval-to-vote binding: when a retrieved state-law family plausibly protects against the local restriction, make the relevant perspective call or explicitly clear it.
- Add or sharpen `Search protocol:` so this perspective searches differently from the others while staying corpus-first and generic-official-source only.
- Clarify how this perspective handles contested legal definitions through perspective-specific interpretation; legal labels are candidate issues, not automatic YES votes.
- Improve citation and retrieval discipline using the available municipal text, state code, corpus search, and web tools when they are genuinely useful.
- Preserve useful unique true positives.
- Reduce shared false positives caused by shared trigger words and repeated reasoning patterns.
- If the target is `jurisdiction_defense`, preserve its antagonistic rare-YES role: defend local authority unless the state-law conflict is clear, directly applicable, and unreconcilable.

Interactive tools. You must call at least one tool before submitting:

- `get_examples(type, n)` — FP (`fp`) or FN (`fn`) examples with municipal text and the perspective's own reasoning.
- `get_pair_detail(state, muni)` — full model output for one (`state`, `muni`) pair.
- `submit_block(block)` — submit the final rewrite. Call this exactly once.

Required workflow.

1. Read the diagnostic: note the FP/FN counts and the "shared vs unique votes" section.
2. Call `get_examples('fn', 5)` to understand what this perspective missed.
3. Call `get_examples('fp', 3)` to understand the worst incorrect flags without over-correcting toward conservatism.
4. Design a rewrite that (a) recovers missed real conflicts, (b) preserves valuable unique true positives, and (c) avoids herd false positives with a distinct legal role and evidence standard.
5. Call `submit_block` with the rewrite.

Rewrite rules.

- Keep the `- **{name}**:` header exactly.
- Rewrite only this perspective block; do not rewrite the whole prompt.
- Keep the block concise but behaviorally specific.
- Include `Search protocol:`, `Vote gate:`, and `Tendency to call conflicts:`.
- The perspective must still find real violations; do not make it vacuously strict.
- Do not make the perspective broadly permissive merely to raise recall.
- Return only the rewritten block, nothing else.

C Compute Budget and Infrastructure

We use the close-sourced model `gemini-3-flash-preview` throughout the paper, and rely on google cloud compute to make API requests. Our total training and inference budget is $100.

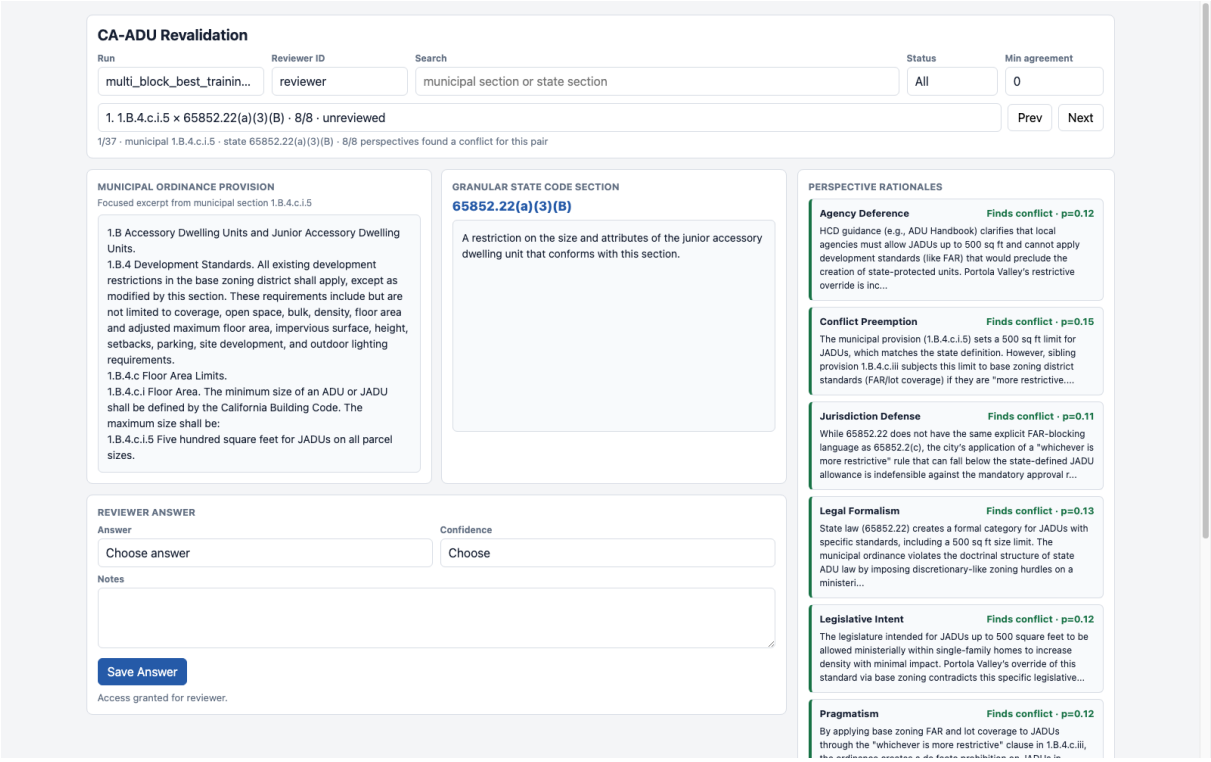

Figure 8: Annotation Dashboard for Our Human Annotators

D Human Annotation Dashboard

We present our annotation dashboard for human annotators in Figure 8.

E AI-assistance

We utilize AI to help us polish writing of the paper, refine experiment code, and plot figures in the paper.