Legal Judgment Prediction: A Reflection on the State of the Art

Yi Feng[1], **Chuanyi Li**[1*], **Vincent Ng**[2]
[1]State Key Laboratory for Novel Software Technology, Nanjing University, China
[2]Human Language Technology Research Institute, University of Texas at Dallas, USA
{fy,lcy}@nju.edu.cn, vince@hlt.utdallas.edu

Abstract

Automatic legal judgment prediction (LJP) has recently received increasing attention in the natural language processing community because of its practical values in the real world. Significant progress has been achieved on LJP in the past decade. However, most existing LJP research primarily focuses on developing methods that achieve better performance on standard evaluation datasets, with limited emphasis on the long-term advancement of the field beyond improving evaluation metrics. In this position paper, we reflect on the state of the art in LJP research, and explore issues that should motivate researchers to think beyond merely enhancing performance metrics, with the ultimate goal of sparking discussions among LJP researchers about the future trajectory of the field.

1 Introduction

Legal Judgment Prediction (LJP) involves predicting judgment outcomes (such as violations, relevant law articles, charges, etc.) based on the fact descriptions of cases (Niklaus et al., 2024; Braun and Matthes, 2024; Xu et al., 2024; Feng et al., 2022b; Chalkidis et al., 2019; Yuan et al., 2026; Shi et al., 2025), possibly supplemented by additional inputs like claims (Malik et al., 2021). In recent years, LJP has attracted significant attention within the AI for Law community.

While LJP has been studied for over 60 years (Kort, 1957), the task remains far from being solved. Nonetheless, LJP has made consistent progress. In recent years, a common approach to LJP has involved using a complex neural model that outperforms competitors on standard evaluation datasets. While improving evaluation metrics is a key short-term goal in LJP research, this focus may not necessarily contribute to the long-term growth and advancement of the field.

In this position paper, we reflect on the state of the art in LJP research, exploring issues that should motivate researchers to think beyond merely enhancing performance w.r.t. predefined metrics. Specifically, as we move from research to practice where LJP systems are being deployed in real-life scenarios, it no longer suffices for LJP systems to yield accurate performance. To improve user confidence in a LJP system, we propose to build next-generation LJP systems that are *trustworthy*. Specifically, trustworthy LJP systems should (1) acquire all relevant legal knowledge necessary for making judgments; (2) follow human reasoning processes rather than relying solely on patterns learned from labeled datasets; and (3) provide clear explanations for both legal professionals and non-professionals, to enable them to understand how a particular conclusion is derived. We hope this paper can spark discussions among LJP researchers about the future trajectory of the field.

2 Related Work

LJP can be broadly defined as the task of predicting the judgment results of a case based on its factual description, such as predicting the applicable law articles, the charges, and the terms of penalty. An overview of existing approaches to these and other LJP tasks can be found in Appendix A. Below we center our discussion on three aspects of work on LJP and related areas that are most relevant to the discussion in the rest of the paper.[1]

2.1 Legal Argument Mining

Given an argumentative text, the goal of *argument mining* is to construct an *argument tree*, which

*Corresponding author

[1]Our goal is *not* to provide a survey of LJP research; rather, we provide the readers with the relevant background in LJP with the goal of facilitating the discussion of the issues in the later sections. For readers interested in a comprehensive overview of LJP research, we refer them to the surveys recently published by Feng et al. (2022a) and Cui et al. (2023b).

Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 45254–45273

Facts: The defendant Wang was involved in a quarrel with his father Li, who was suffering from a terminal illness that had caused prolonged severe pain and physical deterioration. During the verbal dispute, Li expressed distress and frustration, and pushed Wang twice. Moved by deep filial affection, and unwilling to see Li continue to endure intense physical suffering, Wang picked up a kitchen knife and slashed Li in the chest area, targeting the heart. Li collapsed immediately. According to forensic examination, Li sustained a penetrating wound, measuring approximately 4 centimeters in length.
Court View: The Court finds that Wang engaged in a dispute and intentionally used a knife to injure Li, resulting in a second-degree minor injury. His conduct constitutes the crime of intentional injury under Article 234 of the PRC Criminal Law. As to whether the defendant's actions qualify as justifiable defense, the Court holds that under Article 20 of the PRC Criminal Law, justifiable defense requires that the act of defense be in response to an ongoing unlawful infringement and be proportional to the nature of the infringement. In this case, although Li did push Wang during the quarrel, the conduct was of a minor nature and does not constitute a serious or ongoing unlawful infringement. Wang's subsequent act of using a knife to slash Li was clearly excessive, aggressive in nature, and carried retaliatory intent. Therefore, his behavior does not meet the legal requirements for justifiable defense, nor does it qualify as excessive defense. Although the defendant claimed that his conduct was motivated by filial affection and compassion to relieve the victim's enduring illness and pain, such motive does not exempt criminal liability. Considering the unique circumstances, and referring to the precedent (2013) Gui Xing Zhong Zi No. XXX, in which the defendant acted out of filial piety and compassion in causing harm to another, the Court acknowledges the humanitarian motive as a mitigating factor. In light of this precedent and the present case's facts, the Court finds it appropriate to impose a sentence lighter than the statutory average for similar offenses.
Judgment: Under Articles 234 and 20 of the PRC Criminal Law, the defendant Wang is convicted of intentional injury and sentenced to seven months' imprisonment, suspended for one year.

Table 1: Example of a real judgment document for a case judged by civil law.

reveals the argumentative structure of the given text. An argument tree can be defined *recursively*. Specifically, the root node of an argument tree corresponds to the main claim made in the text, and each of its children is a piece of evidence that supports its parent. If a child corresponds to a fact, it no longer needs further support and will therefore have no children (i.e., it is a leaf node). Otherwise, a child must be a subclaim that needs further support, and hence it will be supported by one or more child nodes, each of which will again be either a fact of another subclaim. An *argument* is composed of a claim and all of its supporting evidences. Hence, a non-leaf node and all of its children in an argument tree constitute one argument.

Legal argument mining concerns the automatic construction of argument trees from legal judgment documents. Consider the legal case in Table 1, which is composed of three parts: (1) the *facts* of the case; (2) the *court view*, which is the analysis of the case made by the court judge that explains the rationale behind their judgment decision; and (3) the *judgment* on the case. As can be seen, the court view and the judgment are effectively an argumentative text in the legal domain, which contains the argument the judge used to back up their judgment decision. Hence, an argument tree can be constructed from the court view and the judgment portions of a legal case. This is the primary goal of legal argument mining. While existing works on legal argument mining differ in terms of the argumentation schemes they used to annotate training data and the types of legal cases they worked with (e.g, criminal, divorce) (Freeman and Farley, 1996; Moens et al., 2007; Xu et al., 2020a), the key takeaway is that argument trees can be constructed from different types of legal cases.

To understand the significance of argument mining, consider the argument tree in Figure 1, which is constructed from the legal case in Table 1. For ease of exposition, we label each node with one of five types: (1) P (a precedent case); (2) N (an applicable norm, i.e., law or legal rule); (3) F (a case fact; (4) SC (an intermediate conclusion, henceforth subconclusion); and (5) C (the final conclusion).

The tree makes it clear the reasoning steps that eventually led to the judgment. First, an application of norm N1 to fact F1 led to the subconclusion (SC1) that Wang had clear intent to injure. SC1, when combined with norm N2 and fact F2, led one

Prompt: You are a legal judge with expertise in judicial reasoning under Chinese criminal law. Please carefully analyze the following case and provide: (1) Your prediction of the legal outcome-specifying the charge, relevant article of the Criminal Law, and recommended punishment. (2) A detailed explanation of your reasoning, including references to relevant legal norms where applicable. Case Fact: [the text in the Fact section of the case in Table 1]
Generated Explanation: Charge: Intentional Injury; Relevant Legal Norm: Article 234 of the PRC Criminal Law; Recommended Punishment: Fixed-term imprisonment of of 6–12 months. Wang intentionally used a knife to slash his father Li during a dispute, resulting in a second-degree minor injury. The act falls under the scope of intentional infliction of bodily harm, which is clearly addressed by Article 234. The means involved a high potential for serious harm, supporting the conclusion of criminal culpability. The injury is classified as second-degree minor injury, not serious injury or death. The act was spontaneous and emotionally triggered rather than premeditated. In line with judicial discretion and precedent, a court might impose a fixed-term imprisonment of 6–12 months, possibly with suspension if Wang shows remorse and actively compensates the victim.

Table 2: ChatGPT-generated legal judgment explanation for the legal case in Table 1 and the associated prompt.

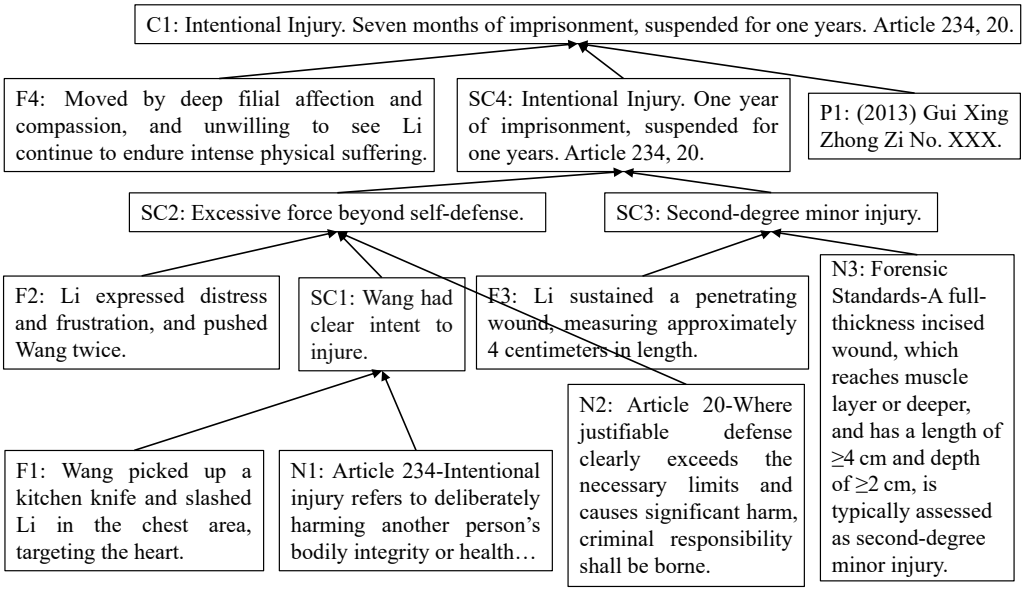

Figure 1: Example argument tree.

to conclude that Wang used excessive force beyond self-defense (SC2). Another norm N3, when applied to fact F3, led to the subconclusion (SC3) that Li's injury is considered second-degree injury. Combining the two subconclusions SC2 and SC3 allowed one to make an initial judgment on this case (SC4): intentional injury with one year of imprisonment suspected for one year. However, after taking into account precedent P1 and fact F4 (the human value that Wang did this out of compassion for his father), the final conclusion C1 is reached, which imposes a lesser term penalty.

2.2 Explanation Generation

One reason that makes existing LJP systems not particularly trustworthy is that the decision *process* is not transparent to humans. Consequently, LJP researchers have begun developing *interpretability* approaches to LJP, where the goal is to provide an *explanation* for the decisions made by an LJP model. For instance, **word- and phrase-based explanation approaches** focus on extracting rationales (typically keywords and phrases) from the facts that serve as evidence snippets to support the legal charge or relevant statutory articles (Feng et al., 2022b; Yue et al., 2021a; Jiang et al., 2018), whereas **sentence-level explanation methods** label full sentences from the fact description that serve as justifications (Malik et al., 2021; Chalkidis et al., 2021). While word/phrase- and sentence-level explanations can present key facts, neither of them present reasoning (e.g., clarifying why the

facts in our example satisfy the legal criteria for intentional injury under Article 234).

Paragraph-level generation methods typically adopt generation frameworks (e.g., seq2seq or LLMs) to generate explanations in the form of natural language paragraphs that not only incorporate fact summaries and legal citations (Deng et al., 2023; Yue et al., 2021b; Vats et al., 2023; Wu et al., 2022; Zhong et al., 2020; Li et al., 2025) but also simulate human judicial reasoning. Perhaps not surprisingly, the best explanation methods thus far are paragraph-level methods that employ LLMs for generation. To enable the reader to understand how good these paragraphs are, we apply ChatGPT-4o to our example case (showing ChatGPT-4o only the facts but not the rest of the case) to make judgment predictions (i.e., charge, term penalty) as well as a paragraph-based explanation of its predictions. The prompt we used and the generated explanation are shown in Table 2.

Although the paragraph-level explanations generated by LLMs for legal decisions are currently the best out there, they are not without their problems, as can be seen from the ChatGPT-generated paragraph in Table 2. By comparing this paragraph and the court view, we can see that this paragraph omitted certain details, such as its failure to use Article 20 to conclude that this was not self-defense. Moreover, despite the fact that ChatGPT outputted the correct predictions, it failed to take into account the pluralistic human values and the precedent when making the predictions. In other words, its reasoning process is partially flawed.

3 A Vision for Legal Reasoning

Next we discuss how legal reasoning processes should be modeled in a trustworthy LJP model.

As noted in the previous section, the explanation provided by ChatGPT-4o omitted several crucial details, and its ability to output the correct legal judgment predictions can be said to be partly due to luck. Of course, LLMs do improve over time, and it is conceivable that they can at some point generate the correct explanation for a legal case. In the ideal situation, LLMs can generate explanations that are as good as the court views.

Nevertheless, even the court view does not necessarily offer the clearest and easiest-to-understand explanation owing to its use of paragraphs as the *representation* of an explanation. As is commonly known, in natural language paragraphs, the *relations* between the sentences or even the text spans of a sentence are not always explicitly expressed (e.g., implicit discourse relations). Similarly, in paragraph-level explanations, the *argumentative* relations that connect the evidences to their corresponding claim are not always explicitly expressed. For instance, in the court view in Table 1, it may not be immediately clear to everyone that the sub-conclusion of a "second-degree minor injury" was supported by the fact "4 centimeters in length" and the applicable Forensic Standards. However, these relations are precisely what enable users, particularly non-legal professionals, to understand which text spans or sentences in a paragraph are involved in a reasoning step in the legal reasoning process.

3.1 Representation

Fortunately, the aforementioned problem associated with a paragraph-based representation is exactly what argument trees can help address, as they reveal the argumentative structure (and hence the reasoning steps) involved in a legal judgment, henceforth the following recommendation:

Recommendation #1: We recommend that legal reasoning be expressed in the form of an *argument tree*, as the structure of an argument tree makes it crystal clear what reasoning steps are involved.

As discussed in Section 2.1, each non-leaf node in an argument tree together with all of its immediate children form an argument and represents exactly one reasoning step, and the order in which these reasoning steps are applied to derive the conclusion is apparent from the tree.

3.2 Learning

We recommend that a *supervised* approach be used to create argument trees. To understand the reasoning logic behind a legal decision, there is nothing better than using the court views on legal cases, in which the reasoning steps behind the judgments are clearly laid out. Specifically, we can first use existing work on legal argument mining (Habernal et al., 2024) to construct an argument tree from each court view. Then, we can use the resulting argument trees as supervisory labels to fine-tune an LLM to enable it to learn how to generate argument trees. Each example used for fine-tuning corresponds to the facts of a case, and its "label" is the corresponding argument tree.

Note that this task of *argument tree generation* is a novel and challenging task. Existing work on legal argument tree mining has focused on argument

tree *extraction* (see Section 2.1), where the goal is to construct an argument tree where the content of all of its nodes are present in the input text. In contrast, in argument tree generation, only the leaf nodes (i.e., facts) of the tree to be generated are given as input, and the rest of the tree, including all of the intermediate conclusions, have to be generated *on the fly*. Fortunately, since court views can be obtained in large amounts easily, we have a lot of data for fine-tuning an LLM.

The success of the above procedure depends on the accuracy of the argument mining tool we used to automatically produce the supervisory labels. If the tool does not produce accurate trees, we can fine-tune it with human-annotated argument trees. We expect that this can be accomplished without a lot of annotations because argument tree extraction is much simpler than argument tree generation.

3.2.1 Decomposing the Learning Task

At first glance, generating a structured output as complex as an argument tree is a daunting task. If it is indeed too difficult to learn a model to output a tree accurately given a legal case, we can try to decompose this complex learning task into a set of smaller, arguably less complex learning tasks via identifying structural characteristics of a *legal* argument tree and understanding the underlying process through which a legal argument tree is produced.

Specifically, when making judgments, a human judge starts by reasoning with the facts of a case, then identify the norms (i.e., laws, legal rules, and precedents) that are applicable to one or more facts to derive a conclusion. If this is an intermediate conclusion (as opposed to the final judgment on the given case), it will be combined with other fact(s) and intermediate conclusion(s) to derive another intermediate/final conclusion. In other words, the human judgment process is incremental.

Motivated by this observation, we propose an *incremental* approach to argument tree construction. Since the human judgment process starts from the facts and the laws/rules/precedents that are applicable to the facts, which correspond to the leaves of an argument tree, we propose to build an argument tree in a bottom-up fashion, via the following steps:

1. Determining the key facts. As seen in Table 1, a legal case is composed of a set of facts. In this example, "the defendant Wang was involved in a quarrel with his father Li" is a fact. A subset of these facts are *key facts*. Specifically, a key fact is a fact that instantiates a precondition of a legal norm: it is a concrete manifestation of a norm's precondition. Note that only key facts can appear in an argument tree: a fact that is not a key fact cannot be combined with a norm by definition and therefore cannot appear in the tree. In our example case, F1, F2, F3, and F4 are therefore key facts.

As our first step, we determine which facts in a given case are key facts. To do so, we begin extracting all the facts from a given case, specifically by (1) treating each clause in the Facts section as a fact, and (2) decomposing each of the resulting facts into atomic facts using an LLM (e.g., "Li sustained a penetrating wound, measuring approximately 4 centimeters in length" can be decomposed into two atomic facts, one composed of the text before the comma and one after).[2] Next, we propose to train a binary classifier to determine whether each of these extracted facts is a key fact or not. Each training instance therefore corresponds to a fact. Since key fact identification requires domain knowledge of what can match the precondition of a norm, we have domain experts label each training instance, whose label is Yes if it corresponds to a key fact and No otherwise.

2. Determining the applicable norms. As noted above, an argument tree has two types of leaves, the *key facts* and the *legal norms* (i.e., laws, legal rules) that are applicable to one or more key facts. In this second step, we propose determine whether a given norm is applicable to a given key fact (or a given subset of key facts) obtained in the previous step via a four-stage *coarse-to-fine* approach.[3]

Stage I: Extract the preconditions from all the norms in our database.[4] Since each norm is already expressed in the form of a set of preconditions, this step is straightforward.

Stage II: Use a lightweight information retrieval approach (such as cosine similarity) to rank all the

[2]We do not derive new facts from other facts or from implicit elements for two reasons. First, the facts of a legal case are already written in such a way that no derivation from other facts or from implicit elements are needed. Second, under well-established adjudicative principles, courts decide cases on facts proven by admissible evidence, not on assumptions that have not themselves been proved.

[3]Note that our approach does *not* assume a one-to-one mapping between a fact and a norm: it allows a single factual premise to be shared by multiple arguments. However, allowing a shared premise to have multiple parents would result in argument graphs rather than trees. To resort to a tree representation, we follow previous work (e.g., Modgil and Prakken (2014)), where we make multiple copies of a given fact. This means that if the fact is being used in N arguments, we create N leaf nodes, each of which contains the same fact.

[4]Given a jurisdiction, its legal norms are readily available.

preconditions extracted in the first step with respect to the key fact, where each precondition and each key fact is expressed as a dense vector. Since this is lightweight, it can be done efficiently.[5]

Stage III: Classify whether a key fact can satisfy a precondition or not. To create the instances to train this binary classifier, we pair a key fact in the training set with each of the top-k preconditions retrieved in Stage II. Test instances are created similarly. Using only the top-k instances can address efficiency concerns. Note that we train a classifier rather than a ranker because a key fact can potentially be matched with more than one precondition, and a classifier enables that flexibility.

Stage IV: From the given set of key facts, create subsets of key facts, and classify whether a fact subset is applicable to a norm. Note that the large number of norms and the large number of subsets of key facts poses an efficiency concern for training and testing this binary classifier. We address this concern as follows. First, recall that in Stage III, we associated each key fact with one or more norms whose precondition it matches (e.g., fact 1 matches with a precondition in norm 1 and norm 3, etc.). We create all and only those subsets of key facts such that all the key facts in each subset share the same applicable norm (note that they can share more than one applicable norm). We then create one training/test instance for the binary classifier from each fact subset and each of the applicable norms they share. This works because there is no point evaluating the applicability of a fact subset against a norm N unless all the facts in the subset can be matched with a precondition in N.

3. Applying the norms. While neither of the tasks in the previous two steps involves generation, our next task does. Specifically, now that we have identified the key facts and determined which norm is applicable to which fact subset, our next task involves applying the norm to the fact subset to generate a conclusion, which we will refer to as a *subconclusion* if it is an intermediate conclusion.

We propose to train a model to generate this (sub)conclusion in a supervised fashion, where the training instances are created as follows. From each argument tree associated with a legal case in the training set, we extract each lowest-level subtree (i.e., the subtree rooted at a node where all of its children are leaves) and create one training instance from each such subtree. (As an example, in the tree in Figure 1, the lowest-level arguments are those rooted at SC1 and SC3.) By definition, each subtree has the fact subset and the applicable norm as the leaves and the (sub)conclusion as the parent. Using the resulting training instances, we can train a seq2seq model that takes as input the fact subset and the applicable norm identified in Step 2, and outputs the (sub)conclusion.

Intuitively, this generation task should not be particularly difficult. Recall that a norm is composed of a set of pre-conditions that need to be satisfied and a consequent that can be derived if the pre-conditions are satisfied. Hence, the sequence-to-sequence model essentially needs to learn to generate a (sub)conclusion that is an instantiation of the consequent with the fact subset.

4. Generating additional subconclusions. The first three steps allow us to generate the lowest-level arguments in an argument tree. In this step, we describe how to complete the tree generation process by deriving additional subconclusions from those we obtained in Step 3. There are two ways in which additional subconclusions can be derived.

First, noting that norms can be applied to not only facts but also subconclusions, we can view the subconclusions obtained in Step 3 as "facts" and augment the case facts with these subconclusions. With an augmented set of facts, more norms may become applicable, and this in turn will produce more subconclusions. (As an example, in the argument tree in Figure 1, N2 is being applied to fact F2 and subconclusion SC1 to produce subconclusion SC2.) Implementation-wise, all we need to do to generate additional subconclusions is to repeat Steps 2 and 3 on this augmented set of "facts".[6]

Second, the subconclusions obtained in Step 3 can be combined with one or more facts and/or subconclusion(s) generated thus far to derive additional subconclusions. Referring to the argument tree in Figure 1, subconclusions SC2 and SC3 together support the derivation of SC4. To perform such derivation, we propose to train a model that

[5]Oftentimes, there is no need to rank a given key fact against all preconditions in practice. For instance, if a key fact is extracted from a divorce case, we should in principle not need to rank against it the preconditions derived from norms concerning murder (unless in the rare cases where murder is involved in the divorce). In other words, we often rank only a subset of the preconditions in practice (using case/law types), making this stage even more efficient than it appears to be.

[6]Recall that the two models in Steps 2 and 3 were originally trained on facts and norms. To make them applicable to subconclusions, all we need to do is to retrain them on additional instances that correspond to not only facts and norms but also subconclusions.

Facts: In early 2019, a concert promoter signed an agreement with an artist to hold a large stadium show on June 15, 2020. The contract specified the stadium venue and date, with ticket sales starting months in advance. In March 2020, due to COVID-19, the state governor issued an executive order prohibiting mass gatherings. The stadium was closed indefinitely, and the ban remained in force on the concert date. The promoter canceled the show and offered refunds; the artist's management claimed breach of contract and sought remaining guaranteed payments, arguing that the cancellation was within the promoter's control.
Judgment: The court examined a contract that required hosting a live, in-person concert at the named venue on the agreed date. It considered whether the pandemic and resulting government orders rendered performance legally and physically impossible. Because the ban on large public gatherings was found to make the core purpose of the contract objectively impossible. Hence, the judges compared this to Taylor v. Caldwell's rule: if the existence or availability of a specific venue is an essential condition and, without fault of either party, that condition is removed (by destruction or lawful closure), therefore the duty to perform is discharged. Here, the government's legal prohibition had the same effect as physical destruction of the venue.

Table 3: Example of a real judgment document for a case judged by common law.

takes any subset of facts and subconclusions as input, and outputs either a subconclusion that can be derived or NIL if nothing can be derived. Instances for training this Derivation model can again be obtained directly from the argument trees corresponding to the training cases: all arguments that do not involve norms can essentially serve as training examples, with the input being the children and the label being the subconclusion in the parent.

Note that generating the right subconclusion given set of facts and subconclusions is by no means an easy task, but we can potentially improve this model using *contrastive* learning, where the positive training examples involve those where the correct subconclusion is being generated, and challenging negative examples can be constructed from the positive training examples by removing one or more input fact/subconclusion.

Recommendation #2: We recommend an incremental, bottom-up approach to argument tree generation. We are by no means claiming that the argument tree generation task can be easily solved with this approach: while the decomposed tasks are easier to perform than generating trees directly, they are by no means trivial. As a position paper, our goal is not to solve the problem, but to provide a viable path to addressing this long-term challenge.

3.3 Cross-Jurisdiction Applicability

In civil law jurisdictions (e.g., France, Germany, China), LJP is statute-centric: judicial decisions are primarily derived from codified legal provisions and principles. Here, an argument tree would place statutory articles in the premise nodes, eventually branching up to the final conclusion. In contrast, in common law jurisdictions (e.g., United States, United Kingdom), LJP is precedent-driven: judicial decisions rely heavily on analogical reasoning from previous cases. In this context, an argument tree would represent key precedents as premises. So far we have described how our approach can be applied to generate argument trees for cases in civil law jurisdictions, a natural question is: what needs to be changed in order for our approach to be applicable to cases in common law jurisdictions?

To answer this question, we make a key observation: the legal cases in the two jurisdictions differ by whether norms or precedents are used in the judgment process. Given this observation, if we can view each precedent as a norm by *representing* a precedent as a set of preconditions together with the corresponding conclusion, we will be able to apply the exact same framework that we described in the previous subsection to generate argument trees for legal cases in common law jurisdictions. Note that being able to do so would allow us to have a *unified* framework for LJP that can elegantly handle legal cases in both jurisdictions.

To see how we can represent a precedent as a norm, consider the legal case in Table 3, which has a Facts section and a Judgment section that shows which precedent was used in the judgment decision. In this example, the preconditions that we want to extract are those of Taylor v. Caldwell's rule, specifically: (1) the existence or availability of a specific venue is an essential condition, and

(2) without fault of either party, that condition is removed (by destruction or lawful closure). In general, preconditions are explicitly enumerated in the Judgment section because this is where the judge needs to explain which precedent is applicable and why. Answering this "why" question entails enumerating why each precondition required for precedent applicability is satisfied. Hence, precondition identification from a precedent case is an extraction problem. There are lexical cues and patterns (e.g., "if... and... therefore") that one can exploit to design heuristics to extract the preconditions. Alternatively, one can employ an argument mining system or an LLM to extract them.

Recommendation #3: We recommend reducing precedents to norms so as to enable the cross-jurisdiction applicability of our approach. Analogous to Universal Dependencies (de Marneffe et al., 2021), which involves modeling dependency structures in a language-independent manner, we believe that our proposal has the potential to enable argument trees to serve as a *jurisdiction-independent* representation for Universal Legal Reasoning.

3.4 Incorporating Pluralistic Values

Legal judgment is not like a "vending machine": humans can make *human-centered* judgments that are rooted in empathy and human connection, rather than being cold and mechanical. For instance, in a murder case, a son who is unwilling to witness his father's suffering from illness may reluctantly choose to end his father's life. In real-world judgments, judges would take into account these emotional factors and consider reducing the sentences (Abrams and Keren, 2010; Welch, 1997). As another example, cases involving racial tensions, such as those related to the Black Lives Matter movement, require judges to evaluate not only objective rules but also the underlying social conflicts and values. Therefore, handling cases cannot be confined to strict legal rules; it must also consider public sentiment and the restoration of social relationships (Martin, 2007). Even if current approaches achieve transparency and interpretability, the absence of human values makes them untrustworthy. To our knowledge, none of the existing methods considers the human *values* that practitioners consider in their reasoning processes.

In the context of LJP, human values refer to the shared ethical and moral principles that AI systems must adhere to during the decision-making process. These include fairness, transparency, accountability, respect for human rights, and the ability to explain decisions in a way that aligns with societal norms and expectations. Such values ensure that AI-driven legal systems not only provide efficient and accurate predictions but also uphold justice and humanity in their operations (Yamane, 2020; Rogers and Bell, 2019; Economou, 2019).

Broadly, human values in LJP include: (1) *values* (intrinsic goods or ideals that people pursue or cherish, e.g., filial piety, justice, self-sacrifice, freedom); for instance, if a son is compelled by filial piety to kill his father, his intention should be regarded as compassion rather than intentional murder; (2) *duties* (moral obligations or responsibilities, e.g., upholding rules, helping others); for example, if a public official sells state secrets, this should be taken into account for harsher punishment; and (3) *rights* (entitlements or claims that individuals have against others or society, e.g., the right to education, healthcare, or free speech); for example, when handling cases related to the Black Lives Matter movement, judges must prioritize the racial conflicts involved and make decisions based on the value of racial equality.

A key advantage of our argument tree generation framework is that it enables human values to be incorporated *naturally* into a LJP model's decision-making process. Consider the case in Table 1. The court view *explicitly* mentioned that the defendant's conduct was motivated by filial affection and compassion to relieve the victim's illness, and that this human value (i.e., filial piety) is being taken into account in the court's judgment. As can be seen in Figure 1, rather than incorporating the human value into the tree, what is incorporated into the corresponding argument tree is a premise that explains how the value is being applied to this case (node F4). We will henceforth refer to this premise as a pluralistic *consideration* to differentiate it from the human value from which it originates. Hence, when we train an argument tree generation model on such cases, the model should already be able to learn how to generate any pluralistic considerations that should be used as part of the tree generation process. This provides further support for our earlier hypothesis that our framework enables argument trees to serve as a jurisdiction-independent representation of the legal reasoning process.

Since the human value from which a pluralistic consideration originates is not directly observable in an argument tree, we propose to label each pluralistic consideration with the corresponding human

value using an LLM, possibly by prompting it using the human values (and their definitions) taken from an existing taxonomy (e.g., Kiesel et al. (2023), Preniqi et al. (2024)). Since the way these abstract, universally applicable human values are interpreted in the legal context may differ across jurisdictions, we propose to induce jurisdiction-specific human values as follows. First, we partition the pluralistic considerations in the training set according to the human values assigned by the LLM, with one set per human value. For each set, we apply a clustering algorithm to induce sub-clusters, and ask an LLM to label each sub-cluster with not only the possibly jurisdiction-specific human value but also a short description of the condition(s) under which this value is applicable. Finally, we ask a legal expert to review the resulting clusters and labels and make iterative refinements as necessary.

Given a legal case l in the test set, we propose to incorporate pluralistic considerations into the argument tree generation process as follows. In Step 1, we generate the list of pluralistic considerations associated with l using a text generator (data for training this generator is readily available since we have extracted for each training case the list of pluralistic considerations). If directly generating the pluralistic considerations given a case is deemed too challenging, we can first label l with the set of jurisdiction-specific human values involved in it using a trained model (data for training this model is also available since all pluralistic consideration in the training set are labeled with these values), and then condition the text generator mentioned above on not only the input legal case but also the human value(s). In Step 2, we train a model that takes as input any subset of the subconclusions we have generated so far for the argument tree that do not yet have parents and any subset of the pluralistic considerations we have generated in Step 1, and outputs a higher-level subconclusion or the root node if the two sets can be combined.

In some legal cases multiple human values may be involved. Note that these values are not absolute: they are weighted against one another when they conflict (Alexy, 2020). For example, certain values are given more importance in a jurisdiction (e.g., freedom of speech in U.S. constitutional law vs. public order in Singapore) (Carrillo, 2023). Fortunately, we can determine how conflicting values are weighted in a given jurisdiction probabilistically. Recall that each pluralistic consideration in each argument tree in the training set has been automatically assigned a human value by the LLM. So, when we see two human values appear in the same tree, we can look at its root node, which contains the judgment decision, to determine which of the two values is given more importance in the judgment process. Aggregating these per-tree statistics over all argument trees in the training set enables us to estimate how different values are weighted against each other when they co-occur. These statistics would be helpful for generating the LJP decisions at the root node of an argument tree given a legal case in the test set.

While incorporating human values into decision making is not new, value alignment approaches developed for the general domains (e.g., rule-based constraints, Reinforcement Learning from Human Feedback, instruction fine-tuning) are insufficient for LJP (see Appendix B for a detailed discussion). Because LJP requires jurisdiction-aware, interpretation-linked, and structurally embedded human value integration, the challenge is qualitatively different from general value alignment in AI. Our proposal addresses this gap by designing a system where human values are formalized as reasoning premises within argument trees, parameterized by jurisdiction.

Recommendation #4: LJP systems should use human values when making decisions. By modeling each pluralistic consideration as a premise in an argument tree, we confine all jurisdiction-specific information within a node, enabling argument trees to continue to serve as a jurisdiction-independent representation of the legal reasoning process.

4 Conclusion and Further Discussion

LJP researchers have primarily focused on enhancing evaluation performance on standard datasets, paying insufficient attention to the long-term challenges in the field. To build next-generation LJP systems that are trustworthy, we laid out a vision for legal reasoning that models the process as argument tree generation, demonstrating that argument trees have the potential to serve as a jurisdiction-independent representation of the legal reasoning process that enables pluralistic values to be naturally incorporated. Further discussion of the issues related to our proposal can be found in the Appendix, including dataset issues (Appendix C), evaluation issues (Appendix D), scalability issues (Appendix E), and the broader applicability of our framework to scenarios beyond LJP (Appendix F).

Limitations

The views expressed in this position paper are necessarily subjective, so readers may potentially disagree with some or all of our proposed recommendations for future research directions. Additionally, we cannot provide a comprehensive overview of the related research in LJP given the space limitations, so a reader without the relevant background may not be able to appreciate our recommendations.

Ethical Considerations

LJP systems should be developed to support, rather than replace, legal professionals in their decision-making processes while providing legal consulting suggestions to individuals with limited legal knowledge. Ultimately, the final decisions should be made by the professionals themselves. While existing LJP approaches have high scores on evaluation metrics, an equally important ethical consideration is whether these systems follow the reasoning processes that human judges would employ when making predictions. In other words, LJP systems should align with human reasoning logic rather than relying solely on patterns learned from datasets.

Acknowledgments

We thank the reviewers for their valuable comments on an earlier draft of this paper. This work was supported by National Natural Science Foundation of China (No. 62406139), State Key Laboratory for Novel Software Technology at Nanjing University (KFKT2025A15, ZZKT2025B14, KFKT2024A07, ZZKT2024B02), Fundamental and Interdisciplinary Disciplines Breakthrough Plan of the Ministry of Education of China (No. JYB2025XDXM118).

References

Kathryn Abrams and Hila Keren. 2010. Who's afraid of law and the emotions. *Minnesota Law Review*, 94:1997.

Nikolaos Aletras, Dimitrios Tsarapatsanis, Daniel Preotiuc-Pietro, and Vasileios Lampos. 2016. Predicting judicial decisions of the European court of human rights: A natural language processing perspective. *PeerJ Computer Science*, 2:e93.

Robert Alexy. 2020. Constitutional rights, balancing, and rationality. In *Habermas and Law*, pages 265–274. Routledge.

Daniel Braun and Florian Matthes. 2024. AGB-DE: A corpus for the automated legal assessment of clauses in german consumer contracts. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2024, Bangkok, Thailand, August 11-16, 2024*, pages 10389–10405. Association for Computational Linguistics.

Piero Ríos Carrillo. 2023. Proportionality, comparability, and parity: A discussion on the rationality of balancing. *Legal Theory*, 29(4):257–288.

Ilias Chalkidis, Ion Androutsopoulos, and Nikolaos Aletras. 2019. Neural legal judgment prediction in english. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 4317–4323. Association for Computational Linguistics.

Ilias Chalkidis, Manos Fergadiotis, Prodromos Malakasiotis, Nikolaos Aletras, and Ion Androutsopoulos. 2020. LEGAL-BERT: The muppets straight out of law school. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2898–2904, Online. Association for Computational Linguistics.

Ilias Chalkidis, Manos Fergadiotis, Dimitrios Tsarapatsanis, Nikolaos Aletras, Ion Androutsopoulos, and Prodromos Malakasiotis. 2021. Paragraph-level rationale extraction through regularization: A case study on European court of human rights cases. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 226–241, Online. Association for Computational Linguistics.

Huajie Chen, Deng Cai, Wei Dai, Zehui Dai, and Yadong Ding. 2019. Charge-based prison term prediction with deep gating network. 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 6362–6367, Hong Kong, China. Association for Computational Linguistics.

Yongming Chen, Miner Chen, Ye Zhu, Juan Pei, Siyu Chen, Yu Zhou, Yi Wang, Yifan Zhou, Hao Li, and Songan Zhang. 2024. Leverage knowledge graph and large language model for law article recommendation: A case study of Chinese criminal law. *arXiv preprint arXiv:2410.04949*.

Kyunghyun Cho, Bart van Merrienboer, Çaglar Gülçehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using RNN encoder-decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1724–1734. ACL.

Pierre Colombo, Telmo Pessoa Pires, Malik Boudiaf, Dominic Culver, Rui Melo, Caio Corro, Andre FT Martins, Fabrizio Esposito, Vera Lúcia Raposo, Sofia Morgado, et al. 2024. Saullm-7b: A pioneering large language model for law. *arXiv preprint arXiv:2403.03883*.

Jiaxi Cui, Zongjian Li, Yang Yan, Bohua Chen, and Li Yuan. 2023a. ChatLaw: Open-source legal large language model with integrated external knowledge bases. *arXiv preprint arXiv:2306.16092*.

Junyun Cui, Xiaoyu Shen, and Shaochun Wen. 2023b. A survey on legal judgment prediction: Datasets, metrics, models and challenges. *IEEE Access*, 11:102050–102071.

Rong Dai, Yonggang Zhang, Ang Li, Tongliang Liu, Xun Yang, and Bo Han. 2024. Enhancing one-shot federated learning through data and ensemble co-boosting. In *Proceedings of the Twelfth International Conference on Learning Representations*, Vienna, Austria. OpenReview.net.

Marie-Catherine de Marneffe, Christopher D. Manning, Joakim Nivre, and Daniel Zeman. 2021. Universal Dependencies. *Computational Linguistics*, 47(2):255–308.

Wentao Deng, Jiahuan Pei, Keyi Kong, Zhe Chen, Furu Wei, Yujun Li, Zhaochun Ren, Zhumin Chen, and Pengjie Ren. 2023. Syllogistic reasoning for legal judgment analysis. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 13997–14009, Singapore. Association for Computational Linguistics.

Nicolas Economou. 2019. Principles for the trustworthy adoption of AI in legal systems: the IEEE global initiative on ethics of autonomous and intelligent systems. In *Proceedings of the First Workshop on AI and Intelligent Assistance for Legal Professionals in the Digital Workplace (LegalAIIA 2019) the 17th International Conference on Artificial Intelligence and Law (ICAIL 2019), Montreal, Canada, June 17, 2019*, volume 2484 of *CEUR Workshop Proceedings*, pages 2–5. CEUR-WS.org.

Yi Feng, Chuanyi Li, and Vincent Ng. 2022a. Legal judgment prediction: A survey of the state of the art. In *Proceedings of the 31st International Joint Conference on Artificial Intelligence and the 25th European Conference on Artificial Intelligence*, pages 5461–5469.

Yi Feng, Chuanyi Li, and Vincent Ng. 2022b. Legal judgment prediction via event extraction with constraints. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2022, Dublin, Ireland, May 22-27, 2022*, pages 648–664. Association for Computational Linguistics.

Kathleen Freeman and Arthur M. Farley. 1996. A model of argumentation and its application to legal reasoning. *Artificial Intelligence Law*, 4(3-4):163–197.

Leilei Gan, Kun Kuang, Yi Yang, and Fei Wu. 2021. Judgment prediction via injecting legal knowledge into neural networks. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*, pages 12866–12874. AAAI Press.

Ivan Habernal, Daniel Faber, Nicola Recchia, Sebastian Bretthauer, Iryna Gurevych, Indra Spiecker genannt Döhmann, and Christoph Burchard. 2024. Mining legal arguments in court decisions. *Artificial Intelligence Law*, 32(3):1–38.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation*, 9(8):1735–1780.

Wanhong Huang, Yi Feng, Chuanyi Li, Honghan Wu, Jidong Ge, and Vincent Ng. 2024. CMDL: A large-scale Chinese multi-defendant legal judgment prediction dataset. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 5895–5906, Bangkok, Thailand. Association for Computational Linguistics.

Hang Jiang, Xiajie Zhang, Robert Mahari, Daniel Kessler, Eric Ma, Tal August, Irene Li, Alex Pentland, Yoon Kim, Deb Roy, and Jad Kabbara. 2024. Leveraging large language models for learning complex legal concepts through storytelling. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2024, Bangkok, Thailand, August 11-16, 2024*, pages 7194–7219. Association for Computational Linguistics.

Xin Jiang, Hai Ye, Zhunchen Luo, Wenhan Chao, and Wenjia Ma. 2018. Interpretable rationale augmented charge prediction system. In *COLING 2018, The 27th International Conference on Computational Linguistics: System Demonstrations, Santa Fe, New Mexico, August 20-26, 2018*, pages 146–151. Association for Computational Linguistics.

Daniel Martin Katz, Michael J Bommarito, and Josh Blackman. 2017. A general approach for predicting the behavior of the supreme court of the united states. *PLOS One*, 12(4):e0174698.

Johannes Kiesel, Milad Alshomary, Nailia Mirzakhmedova, Maximilian Heinrich, Nicolas Handke, Henning Wachsmuth, and Benno Stein. 2023. SemEval-2023 task 4: ValueEval: Identification of human values behind arguments. In *Proceedings of the 17th International Workshop on Semantic Evaluation (SemEval-2023)*, pages 2287–2303, Toronto, Canada. Association for Computational Linguistics.

Fred Kort. 1957. Predicting supreme court decisions mathematically: A quantitative analysis of the "right to counsel" cases. *American Political Science Review*, 51(1):1–12.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. 1998. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324.

Hengzhi Li, Shubin Cai, and Zhong Ming. 2023. Legal judgment prediction incorporating guiding cases matching. In *Natural Language Processing and Chinese Computing - 12th National CCF Conference, NLPCC 2023, Foshan, China, October 12-15, 2023, Proceedings, Part I*, volume 14302 of *Lecture Notes in Computer Science*, pages 511–523. Springer.

Shangyuan Li, Shiman Zhao, Zhuoran Zhang, Zihao Fang, Wei Chen, and Tengjiao Wang. 2025. Basis is also explanation: Interpretable legal judgment reasoning prompted by multi-source knowledge. *Information Processing & Management*, 62(3):103996.

Wan-Chen Lin, Tsung-Ting Kuo, Tung-Jia Chang, Chue-Han Yen, Chao-Ju Chen, and Shou-De Lin. 2012. Exploiting machine learning models for Chinese legal documents labeling, case classification, and sentencing prediction. *International Journal of Computational Linguistics & Chinese Language Processing - Special Issue on Selected Papers from ROCLING XXIV*, 17(4).

Vijit Malik, Rishabh Sanjay, Shubham Kumar Nigam, Kripabandhu Ghosh, Shouvik Kumar Guha, Arnab Bhattacharya, and Ashutosh Modi. 2021. ILDC for CJPE: indian legal documents corpus for court judgment prediction and explanation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 4046–4062. Association for Computational Linguistics.

Jeffrey Martin. 2007. A reasonable balance of law and sentiment: social order in democratic taiwan from the policeman's point of view. *Law & Society Review*, 41(3):665–698.

Tomás Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Proceedings of NeurIPS*, pages 3111–3119.

Sanjay Modgil and Henry Prakken. 2014. The ASPIC+ framework for structured argumentation: A tutorial. *Argument and Computation*, 5(1):31–62.

Marie-Francine Moens, Erik Boiy, Raquel Mochales Palau, and Chris Reed. 2007. Automatic detection of arguments in legal texts. In *The Eleventh International Conference on Artificial Intelligence and Law, Proceedings of the Conference, June 4-8, 2007, Stanford Law School, Stanford, California, USA*, pages 225–230. ACM.

Stuart S. Nagel. 1963. Applying correlation analysis to case prediction. *Texas Law Review*, 42:1006.

Joel Niklaus, Ilias Chalkidis, and Matthias Stürmer. 2021. Swiss-judgment-prediction: A multilingual legal judgment prediction benchmark. In *Proceedings of the Natural Legal Language Processing Workshop 2021*, pages 19–35, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Joel Niklaus, Veton Matoshi, Matthias Stürmer, Ilias Chalkidis, and Daniel E. Ho. 2024. Multilegalpile: A 689gb multilingual legal corpus. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2024, Bangkok, Thailand, August 11-16, 2024*, pages 15077–15094. Association for Computational Linguistics.

Shounak Paul, Pawan Goyal, and Saptarshi Ghosh. 2020. Automatic charge identification from facts: A few sentence-level charge annotations is all you need. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 1011–1022, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Xiao Peng and Liang Chen. 2024. Athena: Retrieval-augmented legal judgment prediction with large language models. *arXiv preprint arXiv:2410.11195*.

Alina Petrova, John Armour, and Thomas Lukasiewicz. 2020. Extracting outcomes from appellate decisions in US state courts. In *Legal Knowledge and Information Systems - JURIX 2020: The Thirty-third Annual Conference, Brno, Czech Republic, December 9-11, 2020*, volume 334 of *Frontiers in Artificial Intelligence and Applications*, pages 133–142. IOS Press.

Vjosa Preniqi, Iacopo Ghinassi, Julia Ive, Charalampos Saitis, and Kyriaki Kalimeri. 2024. Moralbert: A fine-tuned language model for capturing moral values in social discussions. In *Proceedings of the 2024 International Conference on Information Technology for Social Good, GoodIT 2024, Bremen, Germany, September 4-6, 2024*, pages 433–442. ACM.

Justine Rogers and Felicity Bell. 2019. The ethical ai lawyer: What is required of lawyers when they use automated systems? *Law, Technology & Humans*, 1:80.

Jeffrey A. Segal. 1984. Predicting supreme court cases probabilistically: The search and seizure cases, 1962-1981. *American Political Science Review*, 78(4):891–900.

Weijie Shi, Han Zhu, Jiaming Ji, Mengze Li, Jipeng Zhang, Ruiyuan Zhang, Jia Zhu, Jiajie Xu, Sirui Han, and Yike Guo. 2025. Legalreasoner: Step-wised verification-correction for legal judgment reasoning. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2025, Vienna, Austria, July 27 - August 1, 2025*, pages 7297–7313.

Octavia-Maria Sulea, Marcos Zampieri, Mihaela Vela, and Josef van Genabith. 2017. Predicting the law area and decisions of french supreme court cases. In

Proceedings of the International Conference Recent Advances in Natural Language Processing, RANLP 2017, Varna, Bulgaria, September 2 - 8, 2017, pages 716–722. INCOMA Ltd.

Jingyun Sun, Chengxiao Dai, Zhongze Luo, Yangbo Chang, and Yang Li. 2024. LawLuo: A Chinese law firm co-run by LLM agents. *arXiv preprint arXiv:2407.16252*.

Shaurya Vats, Atharva Zope, Somsubhra De, Anurag Sharma, Upal Bhattacharya, Shubham Kumar Nigam, Shouvik Kumar Guha, Koustav Rudra, and Kripabandhu Ghosh. 2023. Llms - the good, the bad or the indispensable?: A use case on legal statute prediction and legal judgment prediction on indian court cases. In *Findings of the Association for Computational Linguistics: EMNLP 2023, Singapore, December 6-10, 2023*, pages 12451–12474. Association for Computational Linguistics.

Pengfei Wang, Yu Fan, Shuzi Niu, Ze Yang, Yongfeng Zhang, and Jiafeng Guo. 2019. Hierarchical matching network for crime classification. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 325–334, Paris, France. ACM.

Pengfei Wang, Ze Yang, Shuzi Niu, Yongfeng Zhang, Lei Zhang, and ShaoZhang Niu. 2018. Modeling dynamic pairwise attention for crime classification over legal articles. In *Proceedings of the 41st International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 485–494, Ann Arbor, MI, USA. ACM.

Bin Wei, Yaoyao Yu, Leilei Gan, and Fei Wu. 2025. An llms-based neuro-symbolic legal judgment prediction framework for civil cases. *Artificial Intelligence and Law*, pages 1–35.

D. Don Welch. 1997. Ruling with the heart: Emotion-based public policy. *Southern California Interdisciplinary Law Journal*, 6:55.

Di Wu, Jun Bai, Yiliao Song, Junjun Chen, Wei Zhou, Yong Xiang, and Atul Sajjanhar. 2024. Fedinverse: Evaluating privacy leakage in federated learning. In *Proceedings of the Twelfth International Conference on Learning Representations*, Vienna, Austria. OpenReview.net.

Yiquan Wu, Yifei Liu, Weiming Lu, Yating Zhang, Jun Feng, Changlong Sun, Fei Wu, and Kun Kuang. 2022. Towards interactivity and interpretability: A rationale-based legal judgment prediction framework. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, EMNLP 2022, Abu Dhabi, United Arab Emirates, December 7-11, 2022*, pages 4787–4799. Association for Computational Linguistics.

Yiquan Wu, Siying Zhou, Yifei Liu, Weiming Lu, Xiaozhong Liu, Yating Zhang, Changlong Sun, Fei Wu, and Kun Kuang. 2023. Precedent-enhanced legal judgment prediction with LLM and domain-model collaboration. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, EMNLP 2023, Singapore, December 6-10, 2023*, pages 12060–12075. Association for Computational Linguistics.

Chaojun Xiao, Xueyu Hu, Zhiyuan Liu, Cunchao Tu, and Maosong Sun. 2021. Lawformer: A pre-trained language model for Chinese legal long documents. *AI Open*, 2:79–84.

Chaojun Xiao, Haoxi Zhong, Zhipeng Guo, Cunchao Tu, Zhiyuan Liu, Maosong Sun, Yansong Feng, Xianpei Han, Zhen Hu, Heng Wang, and Jianfeng Xu. 2018. CAIL2018: A large-scale legal dataset for judgment prediction. *arXiv preprint arXiv:1807.02478*.

Huihui Xu, Jaromír Savelka, and Kevin D. Ashley. 2020a. Using argument mining for legal text summarization. In *Legal Knowledge and Information Systems - JURIX 2020: The Thirty-third Annual Conference, Brno, Czech Republic, December 9-11, 2020*, volume 334 of *Frontiers in Artificial Intelligence and Applications*, pages 184–193. IOS Press.

Nuo Xu, Pinghui Wang, Long Chen, Li Pan, Xiaoyan Wang, and Junzhou Zhao. 2020b. Distinguish confusing law articles for legal judgment prediction. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3086–3095, Online. Association for Computational Linguistics.

Shanshan Xu, T. Y. S. S. Santosh, Oana Ichim, Barbara Plank, and Matthias Grabmair. 2024. Through the lens of split vote: Exploring disagreement, difficulty and calibration in legal case outcome classification. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2024, Bangkok, Thailand, August 11-16, 2024*, pages 199–216. Association for Computational Linguistics.

Nicole Yamane. 2020. Artificial intelligence in the legal field and the indispensable human element legal ethics demands. *Georgetown Journal of Legal Ethics*, 33:877.

Weikang Yuan, Kaisong Song, Zhuoren Jiang, Junjie Cao, Yujie Zhang, Chengyuan Liu, Jun Lin, Ji Zhang, Kun Kuang, and Xiaozhong Liu. 2026. A multi-agent framework with legal event logic graph for multi-defendant legal judgment prediction. *Information Processing & Management*, 63(2):104319.

Linan Yue, Qi Liu, Binbin Jin, Han Wu, Kai Zhang, Yanqing An, Mingyue Cheng, Biao Yin, and Dayong Wu. 2021a. NeurJudge: A circumstance-aware neural framework for legal judgment prediction. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 973–982, Virtual Event. ACM.

Linan Yue, Qi Liu, Han Wu, Yanqing An, Li Wang, Senchao Yuan, and Dayong Wu. 2021b. Circumstances enhanced criminal court view generation. In

Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, pages 1855–1859, Virtual Event. ACM.

Rong-zhen Zhang, Xiao-yan Meng, Xiao-xiao Liu, and Yang Wang. 2023. Construction of knowledge graph for animal husbandry laws and regulations in China. *Animal Husbandry and Feed Science*, 44(3):69–74.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2020. Bertscore: Evaluating text generation with BERT. In *Proceedings of the 8th International Conference on Learning Representations*, Addis Ababa, Ethiopia.

Yuan Zhang, Wanhong Huang, Yi Feng, Chuanyi Li, Zhiwei Fei, Jidong Ge, Bin Luo, and Vincent Ng. 2024. LJPCheck: Functional tests for legal judgment prediction. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 5878–5894, Bangkok, Thailand. Association for Computational Linguistics.

Haoxi Zhong, Zhipeng Guo, Cunchao Tu, Chaojun Xiao, Zhiyuan Liu, and Maosong Sun. 2018. Legal judgment prediction via topological learning. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3540–3549, Brussels, Belgium. Association for Computational Linguistics.

Haoxi Zhong, Yuzhong Wang, Cunchao Tu, Tianyang Zhang, Zhiyuan Liu, and Maosong Sun. 2020. Iteratively questioning and answering for interpretable legal judgment prediction. In *Proceedings of the Thirty-Fourth AAAI Conference on Artificial Intelligence*, pages 1250–1257, New York, NY, USA.

Zhi Zhou, Jiang-Xin Shi, Peng-Xiao Song, Xiaowen Yang, Yi-Xuan Jin, Lan-Zhe Guo, and Yufeng Li. 2024. LawGPT: A Chinese legal knowledge-enhanced large language model. *CoRR*, abs/2406.04614.

A Current State of LJP Research

In this section, we discuss the commonly used LJP corpora and the salient approaches to LJP. For ease of exposition, we present information on corpora and approaches in Tables 4 and 5 respectively. For each of the corpora, we discuss the jurisdiction for which it was developed, its goal and annotations. Moreover, we describe the general strengths and limitations of existing corpora. As for approaches, we first classify them as rule-based, traditional neural-based, or neural approaches. We then subcategorize neural approaches based on whether they are reasoning-based approaches, interpretability approaches, or approaches that incorporate external knowledge. For each approach, we discuss its strengths and weaknesses.

B Inadequacy of General-Domain Value Alignment Approaches for LJP

In LLMs and general NLP systems, value alignment is commonly implemented through a combination of *rule-based constraints* (manually crafted safety rules to filter or block content that violates broadly agreed ethical norms), *instruction fine-tuning* (training on datasets labeled by annotators to encourage outputs that reflect a target set of social principles), and *RLHF* (learning a reward function based on human preference rankings to steer generation toward "aligned" responses). These approaches work well in general domains because they assume a relatively stable, context-agnostic set of high-level values, and they typically operate either globally at model-training time or as post-hoc output filtering. However, these value alignment approaches that work for the general domains are insufficient for LJP, for the following reasons.

First, in LJP, embedding human values is not a matter of adding general ethical filters. It is an inherent part of the LJP reasoning process that has several domain-specific complexities.

Second, legal values vary fundamentally between jurisdictions. For example, racial equality and freedom of speech carry constitutional priority in U.S. case law, while social harmony and moral education carry much higher weight in Chinese jurisprudence. A single static alignment model cannot capture such variability. Instead, LJP requires value parameterization (represented as a premise node) so that the model can dynamically switch the value set according to the applicable legal system.

Third, in our argument tree generation framework, human values are embedded as premise nodes in legal argument trees, meaning they directly influence the path from evidence to decision. This is fundamentally different from the post-generation filtering or latent preference weighting seen in general-domain AI. Here, value alignment has to be explainable, traceable, and legally justifiable.

Fourth, many values in law are not abstract ideals but legally defined and interpreted through legislation, precedent, and doctrines. Capturing this requires combining value nodes with statutes and precedents, something general-domain alignment methods do not do.

Finally, courts often face conflicts between competing values (e.g., freedom of expression vs. national security) and resolve them using jurisdiction-

Description	Strengths	Weaknesses
Heuristic Approaches		
Modeling legal reasoning as a series of heuristic rules (Kort, 1957; Segal, 1984; Nagel, 1963), each of which is represented in IF-THEN format like "A if B_1, and B_2, ...B_n ". If the case facts align with the preconditions of the rules, the corresponding conclusion is triggered.	Decision is easily interpretable	These rules are often (1) overly simple and (2) too rigid to be generalizable.
Machine Learning Approaches		
Feature-based: cast LJP as a classification task (where judgment outcomes are class labels) and take an off-the-shelf learning algorithm (e.g., Support Vector Machines) to train a classifier (Sulea et al., 2017; Katz et al., 2017; Aletras et al., 2016; Lin et al., 2012), typically using word n-grams as representations of the case facts.	Obviates the need to manually design decision rules.	The semantic representations of a case (using n-grams) are weak.
Neural-based Approaches		
Early models predict judgment outcomes by (1) embedding case facts using methods like Word2Vec (Mikolov et al., 2013), (2) encoding these embeddings with neural encoders such as CNNs (LeCun et al., 1998), LSTMs (Hochreiter and Schmidhuber, 1997), or GRUs (Cho et al., 2014) to capture contextual semantics, and (3) feeding the representations into feedforward classifiers to output task-specific legal labels (Wang et al., 2018; Chen et al., 2019; Paul et al., 2020).	Employs an enhanced semantic representation of a case	Coverage of legal knowledge is low
Reasoning-based Approaches: enable LJP models possess reasoning abilities.		
Legal LLMs and **Legal-specific pre-trained language models** have been developed (Colombo et al., 2024; Cui et al., 2023a; Sun et al., 2024; Xiao et al., 2021; Shi et al., 2025; Yuan et al., 2026), such as Lawyer LLaMA, ChatLaw, Legal-BERT, Lawformer. For English LJP, Legal-BERT has been pre-trained on legal texts from diverse sources, such as legislation, court cases, and contracts (Chalkidis et al., 2020). For Chinese LJP, LaWGPT has been pre-trained on Chinese civil and criminal legal documents (Zhou et al., 2024). Instruction strategies are used to conduct LJP (Vats et al., 2023; Wu et al., 2023; Jiang et al., 2024), such as prompting LLMs to predict judgment outcomes based on facts, candidate labels, and relevant precedent cases (Wu et al., 2023). **Neural-symbolic LJP models** combine semantic representations from neural networks with logical reasoning based on symbolic rules (Gan et al., 2021; Wei et al., 2025). These symbolic rules are typically organized as logical expressions or knowledge graphs. During the prediction phase, the model often adopts a late fusion strategy, integrating the outputs of the neural and symbolic components through weighted combination.	LLMs possess lots of legal knowledge and can understand complex instructions. Symbolic components (such as rule bases or logical representations) can provide explicit reasoning paths, enabling the model to exhibit stronger legal logic rather than relying solely on statistical patterns in the data.	LLMs often generate hallucinations and lack alignment with human legal reasoning, as they depend on implicit patterns from large datasets rather than structured legal logic. This limits their effectiveness in complex or nuanced cases. While integrating neural and symbolic components offers a path toward reasoning, current approaches—such as loss-based fusion of prediction probabilities—are shallow, and predefined rules usually support only single-step inference, hindering the modeling of multi-step legal reasoning.
Knowledge: incorporate various forms of external legal knowledge.		
Dependencies among the judgment results are exploited by (Xu et al., 2020b; Wang et al., 2018, 2019; Zhong et al., 2018): if a law article is predicted, the corresponding predicted term of penalty should fall within the range defined by that law article, rather than exceeding it. By establishing these dependencies among judgments, the model can generate non-contradictory outcomes. Also, **Historical cases** are retrieved as judgment references (Li et al., 2023).	Broadens the range of reasoning rules available to the model but also guides its predictive behavior by referencing the injected reasoning rules	Many models rely on static legal knowledge bases (e.g., statutes, precedents), which may not reflect recent legal updates. Models often incorporate only limited types of knowledge (e.g., statutes), ignoring other critical sources such as case interpretations, procedural context, or expert reasoning, resulting in shallow or biased predictions.
Interpretability: accompanying a model's prediction with a natural language explanation of the prediction.		
Pre-explanation strategies generate explanations before making predictions, such as extracting rationales from case facts to guide legal article prediction (Chalkidis et al., 2021; Zhong et al., 2020; Deng et al., 2023). In contrast, **post-explanation** strategies produce explanatory text after predictions to clarify the reasoning, for example, identifying key sentences whose removal significantly impacts model performance as explanations (Malik et al., 2021; Yue et al., 2021b; Li et al., 2025).	Provides clear explanations to increase trust and acceptance.	The explanations provided by the model are often too technical for ordinary people to understand. Additionally, the reasoning behind these explanations is usually straightforward and lacks the depth needed to cover complex legal cases, making it hard for users to fully grasp the decision process.

Table 4: Approaches to Legal Judgment Prediction.

Dataset/Language	Annotations	Goal
ECHR2021/English (Chalkidis et al., 2021)	(alleged) law articles and violations with paragraph-level rationales	Evaluating models performance on law article prediction and explanations
CAIL2018/Chinese (Xiao et al., 2018)	law articles, charges and prison terms	Evaluating models performance on law article, charge and prison term predictions.
CMDL/Cinese (Huang et al., 2024)	multi-defendant law articles, charges and prison terms	Evaluating models performance on multi-defendant law article, charge and prison term predictions.
LJPCHECK/Chinese (Zhang et al., 2024)	testing instances with judgments, fact elements	Evaluating model performance in extracting factual elements and identifying biases related to ethnicity, location, and other dimensions.
SJP/Multi-lingual (Niklaus et al., 2021)	court decisions	Evaluating models performance on court decision prediction.
ILDC/English (Malik et al., 2021)	court decisions with sentence-level explanations	Evaluating models performance on court decision prediction.
Strengths: Many datasets contain tens or hundreds of thousands of real-world legal cases, enabling robust training and benchmarking of models. These datasets are collected from real legal systems (e.g., Chinese court judgments), making them highly relevant and useful for applied legal AI tasks.		
Weaknesses: Most existing LJP datasets are derived from court judgment documents, which often contain indicative phrases (e.g., "the case is particularly serious") that explicitly hint at the outcome. Models tend to learn these superficial shortcuts rather than genuine legal reasoning, which deviates from real-world application scenarios—where such indicative language is typically absent. While some datasets include explanation annotations, these are often either overly technical or lack sufficient detail, resulting in inadequate support for modeling robust legal reasoning.		

Table 5: Commonly used corpora for Legal Judgment Prediction.

specific balancing tests or proportionality frameworks. General-domain alignment techniques, which operate on static preferences, lack the capability to model such structured, context-sensitive trade-offs.

Because LJP requires jurisdiction-aware, interpretation-linked, and structurally embedded human value integration, the challenge is qualitatively different from general value alignment in AI. Our proposal addresses this gap by designing a system where human values are formalized as reasoning premises within argument trees, parameterized by jurisdiction, and combined to the relevant legal norms. This ensures both legal validity and adaptability, a combination not addressed by existing alignment research.

C Dataset Issues

While the main text focuses on modeling issues, in this section, we discuss the relevant dataset issues.

C.1 Annotation

To train the models for the various tasks we proposed in the previous section, we have suggested using automatically created data or methods that rely on a small amount of labeled instances. While these methods offer a quick way of obtaining data for model training, in the long run it would be desirable to construct labeled datasets to train these models. Beyond training, we also need labeled data for model evaluation. Below we discuss what annotations we propose to create manually.

Argument trees. We proposed to learn how to generate argument trees as a supervised learning task, where the "label" associated with a legal case is its argument tree. To obtain these argument tree "labels", recall that we proposed to use existing argument mining tools to automatically *extract* argument trees from court decisions on legal cases. Hence, the quality of the resulting trees depends entirely on the accuracy of the tools being used. However, even for languages for which legal argument mining is reasonably mature (e.g., English), argument mining F-scores do not exceed 0.9, meaning that these trees are not perfect. For many other languages, such argument mining tools simply do not exist. Hence, there is a need for manual argument tree annotations for training better argument mining tools or building such tools from scratch if they do not already exist.

So far, different legal argument mining researchers have chosen to annotate argument trees on different corpora using different argumentation schemes. We believe that it would be desirable for the LJP research community to develop a shared vision of how this annotated corpus should be developed, including issues as fundamental as which corpus to annotate using which argumentation scheme, as the lack of standardization could hinder research progress in this field. For instance, if different researchers evaluate their models on their own corpora, it makes it difficult to directly compare models and track research progress, and if different argumentation schemes are used to annotate argu-

ment trees, it makes it harder to train models on all available corpora.

Pluralistic values. As mentioned in Section 3.3.1, the annotation of pluralistic values can be performed as part of argument tree annotation.

Recommendation #5: We recommend that legal corpora be annotated with argument trees and human values, and the community develop a shared vision of how annotated corpora should be developed. The LJP community is lacking shared infrastructure and resources, and the development of such corpora would benefit the field in the long run.

C.2 Knowledge Acquisition

To perform legal reasoning, a LJP system needs to be equipped with legal knowledge. Moreover, since legal knowledge is constantly evolving, we need to provide a LJP system with up-to-date knowledge.

While there exist public legal databases (e.g., PKULAW (Zhang et al., 2023)) from which legal knowledge can be acquired, their data may be outdated as legal knowledge evolves over time. Hence, many existing LJP systems search for legal knowledge on the Web (Chen et al., 2024; Peng and Chen, 2024). While open-source data sources offer a wealth of information, identifying authentic content is challenging due to the vast amount of inaccurate as well as fabricated knowledge available.

Recommendation #6: We recommend that a reliable legal knowledge base (KB) be built that contains up-to-date legal knowledge (i.e., legal regulations and historical cases). To ensure that the legal knowledge in the KB is accurate, we can assert that a piece of knowledge cannot be inserted into the KB unless it (1) is downloaded from authentic websites (e.g., government websites) or (2) has been manually verified by trusted parties. We therefore recommend that this KB be built via a collaborative effort by LJP researchers and other stakeholders who care about the importance of trustworthiness in LJP. To ensure that the knowledge in the KB is up-to-date, these trusted parties will have to take the responsibility to maintain the KB by deleting obsolete knowledge and update it with new knowledge. Moreover, some legal data is sensitive, so techniques are required to safeguard legal data privacy. We recommend using federated learning (Wu et al., 2024; Dai et al., 2024), which can alleviate the data privacy problem, to handle sensitive data. We believe that this legal KB is an infrastructure that can have a significant impact on the LJP research community in the long run, so the community should develop a shared vision of how this KB should be built and maintained.

D Evaluation Issues

Among the models we proposed in Section 3, argument persuasiveness, as a regression task, can be evaluated using standard metrics for regression such as the mean squared loss; and clarification generation, as a text generation task, can be evaluated using generation metrics that can capture semantics, such as BERTScore (Zhang et al., 2020).[7]

Evaluating argument trees is less trivial. Specifically, we borrow ideas from the Argument Mining community, where a tree is evaluated in terms of (1) *argument component extraction* performance, where the extraction performance of each type of argument components (e.g., claims, premises) is evaluated; and (2) relation extraction performance, which evaluates the quality of the links between different argument components (e.g., whether a child indeed supports its parent). Both tasks are evaluated in terms of recall, precision, and F-score.

There is a caveat, however. Unlike in argument tree extraction where the tree nodes correspond to text spans extracted from text, in argument tree *generation*, the non-leaf nodes are all *generated*. Hence, when evaluating these nodes, we have to rely on generation metrics such as BERTScore.[8]

Recommendation #7: We recommend examining whether there is any relationship between the persuasiveness scores and the argument tree evaluation scores described above. If so, we can investigate methods for predicting the argument tree scores directly from the persuasiveness scores, thus enabling evaluation to be conducted *without* using (costly-to-obtain) gold argument trees.

E Scalability Issues

In this section, we discuss the scalability issues surrounding our proposed method.

While the space of argument trees results in a combinatorial explosion, it does not necessarily imply that our method is not scalable. Specifically, many problems in AI and machine learning,

[7] Note that these are preliminary proposals and should be modified based on their effectiveness. As a last resort, human evaluation can be employed.

[8] Since pluralistic values are integrated into an argument tree as tree nodes, we can evaluate whether the right values are being used and whether they are used in the right places in the tree, both in terms of recall and precision.

like our argument tree generation task, have large search spaces, yet the search algorithm can be scalable for searching for the hypothesis using, for instance, dynamic programming or heuristics. As a specific example, consider decision tree learning. The possible space of decision trees is exponential in the number of attributes, yet the well-known ID3 decision tree learning algorithm, which uses Information Gain or Gain Ratio as the split criterion, enables a decision tree to be learned efficiently in an incremental, greedy fashion. As for our method, we believe that it is scalable to the legal corpora that are commonly used in the NLP community, for the following reasons.

First, it is piecemeal in nature, building an argument tree in an incremental (bottom-up) fashion. In other words, we are not considering all the possible trees in each step of the tree generation process. This is similar to the classical decision tree learning problem mentioned above: the space of decision trees is huge, but decision tree algorithms such as ID3 operate by building a tree in an incremental (top-down) fashion to achieve scalability.

Second, each step in our approach (including key fact identification and applicable norm identification) can handle all least the commonly used LJP corpora in the NLP community without scalability issues. Specifically, empirical analyses of existing large-scale legal datasets used in the NLP community (Xiao et al., 2018; Petrova et al., 2020; Cui et al., 2023b; Chalkidis et al., 2019) show that the vast majority of court decisions range from 4 to 25 pages, with exceptionally complex cases rarely exceeding 50 pages, and the average number of key fact candidates in well-known datasets (e.g., CAIL2018 (Xiao et al., 2018) and ECtHR (Chalkidis et al., 2019)) is between 15 and 35 key facts per case. It is important to note that while the number of candidate facts can be large, the actual number of key facts is small (i.e., 15-35). Note that all the subsequent steps in our proposal operate on key facts, not candidate facts. In other words, the number of key facts is not going to pose problems with our approach. Since the candidate facts are being used in the key fact extraction step (by the binary classifier) only, even if we have tens of thousands of candidate facts given a case description, this is not going to present any computational issues for our approach.

While it is conceivable that our approach may still not be able to efficiently handle a 1000-page case description where the number of actual key facts is large, we are not claiming that our approach can solve all the challenging problems. Nevertheless, the fact that our approach is able to handle existing legal corpora commonly used in the NLP community represents a good starting point.

Recommendation #8: Further research is needed to address potential scalability issues surrounding our argument tree generation framework. There are many ways in which efficiency can be improved. For instance, legal norms and precedents can be clustered based on semantic type (e.g., norms regarding divorce, murder, robbery), and each precondition associated with a norm/precedent can be assigned a semantic type (e.g., severity of injury, amount of theft), so that key facts can be matched against norms/preconditions in an efficient manner via semantic types and the clustering results. Note that clustering and semantic typing of norms, precedents, and preconditions can be done only once and in an offline fashion, there are certainly ample opportunities to make the argument tree generation process scalable to cases with a large number of facts.

F Broader Applicability

In this section, we discuss the broader applicability of the argument tree generation task.

F.1 Practical Utility

We begin by discussing the practical utility of the argument tree generation task, with the goal of highlighting how the task can help lawyers, judges and legal experts in day-to-day life. Specifically, we provide six example scenarios in which the task would be of practical utility.

First, the argument tree generation task is capable of identifying which legal norms are applicable to a fact. This functionality can assist lawyers and judges in predicting the relevant norms for a case and in offering informed judgment advice when dealing with new cases.

Second, the task can serve as a valuable tool for judges when reviewing their written judgments. In practice, once a case is decided, judges must compile all judicial reasoning processes into a formal legal document. By applying the argument tree-generation task, it is possible to automatically produce a structured, tree-like representation of the reasoning — capturing key facts, applicable norms, conclusions, and the logical links among them. Judges can then compare this generated argu-

ment tree with their own written analysis, allowing them to assess whether their reasoning is legally compelling, identify any gaps or weaknesses, and ultimately enhance the quality and persuasiveness of case documents.

Third, the task also offers law students a practical learning framework and clear analytical guidelines for developing persuasive legal arguments. Typically, when presented with a new case, students attempt to reason through the facts and applicable laws, yet may be uncertain whether their reasoning is accurate or effective, often relying on expert feedback. With the argument tree-generation approach, students can directly compare their own reasoning structures with a generated tree, gaining insight into how well their arguments align with legally sound reasoning patterns and receiving guidance on how to strengthen them.

Fourth, the argument trees generated via this task can be used to enhance historical case retrieval systems. Given a legal case at hand, the goal of historical case retrieval is to retrieve from a historical case base those legal cases that are most similar to the case at hand. Currently, historical case retrieval systems operate by (1) converting each case in the historical case base as well as the case at hand into a dense representation, and then (2) computing the similarity of the dense representations to retrieve the most similar historical cases. This approach sometimes ignores subtle lexical cues in a case that are crucial to accurate historical case retrieval. We can use generated argument trees to improve the retrieval process by (1) representing each case (both the case at hand and each of the historical cases) as an argument tree, and (2) computing the similarity of the argument trees instead. This can potentially improve the accuracy of the retrieval process because (1) similar cases should have similar argument structures (and hence similar trees), and (2) the tree structures can amplify the differences between lexically similar cases that turn out to be semantically dissimilar, thus reducing the chance of retrieving lexically similar but semantically dissimilar cases.

Fifth, the task can support AI-assisted legal drafting and pre-trial preparation. Before a trial begins, lawyers often need to draft briefs that outline the facts, legal norms, and anticipated arguments. By automatically generating an argument tree from the case description, lawyers can obtain a clear, structured map of potential lines of reasoning, including possible counterarguments, and logical dependencies. This not only accelerates the drafting process but also helps identify weak arguments ahead of time, improving preparedness and the overall persuasiveness of submissions.

Sixth, the task can help norm applicability checking, which refers to the process of verifying whether the legal norms applied in a case are appropriate given the established key facts. In the context of reviewing historical cases, legal experts can systematically examine whether the generated norms align with those applied in the original case. Such analysis enables the detection of potential misapplications or omissions of relevant statutory provisions.

F.2 Applicability to Pleas

Recall that our work targets LJP, which operates on fact descriptions, not pleas. While pleas are outside the formal LJP definition, for researchers who are primarily interested in pleas, we believe that our framework can be extended to handle pleas.

Specifically, a plea contains both the case facts as well as the corresponding legal argument. If our approach is applied to a plea, it will generate an argument tree using the facts it contains and ignore the legal argument. Having said this, our approach can be more or less applied to pleas as is, except for plea preprocessing and key fact identification.

To exemplify, consider a plaintiff's plea that "On 10 February 2023, the Defendant failed to deliver the 500 units of medical equipment as specified in Purchase Order #4521, despite having received full payment in advance on 15 January 2023. Under Article 45 of the Commercial Contract Law, a seller must deliver goods within the agreed period after payment has been made.The Defendant's failure to deliver constitutes a material breach of contract. The Plaintiff respectfully requests that the Court order the Defendant to refund the full purchase price of USD 250,000 and compensate for consequential damages in the amount of USD 50,000 pursuant to Article 65 of the same law."

First, pleas are often lengthy and contain irrelevant material, so we first extract or infer candidate facts from the plea. Some facts may be implicit rather than explicitly stated and must be inferred. We therefore generate the full set of fact candidates. After this step, we can obtain a pool of candidate facts for our example plea:

- CF1: Full payment of USD 250,000 was made on 15 Jan 2023.

- CF2: Agreement specified delivery of 500 units by 10 Feb 2023.
- CF3: Defendant failed to deliver goods by deadline.
- CF4: Article 45 imposes delivery obligation after payment.
- CF5: Article 65 provides for refund + consequential damages for breach.

Next, we need to identify the subset of the candidate facts that correspond to key facts (legally relevant and evidence-backed):

- KF1: Full payment of USD 250,000 was made on 15 Jan 2023.
- KF2: Defendant failed to deliver goods by deadline.

Unlike traditional fact description inputs in LJP, pleas may contain no evidence-backed irrelevant information. However, current models cannot check them automatically as fact checking is done by legal experts in practice. Note that while there can be a large number of candidate facts, key fact identification typically yields a small, bounded set to work with to solve the long text problem.

The next step involves determining applicable norms. This step is the same as what we described in the paper (i.e., no change is needed). Note, however, that while the party provides the norms (as part of the legal argument), our approach ignores them and will identify the applicable ones. N1: Article 45 – Delivery obligation: Seller must deliver goods within agreed period after payment. For example, we may get two norms:

- N1: Article 45 – Delivery obligation: Seller must deliver goods within agreed period after payment.
- N2: Article 65 – Remedy for breach: Refund + consequential damages for breach of contract

The next step involves applying the norms to facts to generate conclusions. This step is also the same as what we described in the paper (i.e., no change is needed). We match the key facts to norm preconditions, then reason toward conclusions.

There are two level conclusions. First, KF1 (Payment made) + KF2 (No delivery) + N1 (Article 45) $\rightarrow$ C1 (Defendant is in material breach of contract.) Second, C1 (Material breach+ N2 (Article 65) $\rightarrow$ C2 (Plaintiff entitled to refund of USD 250,000 and consequential damages of USD 50,000.)

Recommendation #9: While our argument tree framework is originally developed specifically to model the legal reasoning process with the goal of enabling LJP systems to make decisions that are trustworthy and transparent, we believe that this framework has broader practical utility beyond LJP. We recommend that researchers look for novel application scenarios to which this framework can be applied.