Legal Judgment Prediction via Event Extraction with Constraints

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@smail.nju.edu.cn, lcy@nju.edu.cn, vince@hlt.utdallas.edu

Abstract

While significant progress has been made on the task of Legal Judgment Prediction (LJP) in recent years, the incorrect predictions made by SOTA LJP models can be attributed in part to their failure to (1) locate the key event information that determines the judgment, and (2) exploit the cross-task consistency constraints that exist among the subtasks of LJP. To address these weaknesses, we propose *EPM*, an Event-based Prediction Model with constraints, which surpasses existing SOTA models in performance on a standard LJP dataset.

1 Introduction

Legal Judgment Prediction (LJP) is a crucial task in the legal judgment decision making process. Given the facts of a legal case, the goal is to predict the court's outcome. So far, English LJP has focused on predicting law articles (Chalkidis et al., 2019a, 2021) and court decisions (Malik et al., 2021) while French LJP (Sulea et al., 2017b) has focused on predicting court rulings. In this paper, we examine LJP in the context of Chinese via the widely used CAIL dataset (Zhong et al., 2018; Xu et al., 2020), which involves three subtasks: predicting (1) law articles, (2) charges and (3) terms of penalty, as shown in Figure 1.

While state-of-the-art (SOTA) LJP models have several fundamental limitations (Binns, 2019), one of the technical issues they face concerns their failure to locate the key event information that determines the judgment results. Consider Figure 2, where the fact statement of a robbery case involves the illegal break-in description. Existing models wrongly predict that the law article is about illegal search since many words describe the break-in process even though the main point is about robbery.

How can we address this problem? Recall that in the continental judicial system, a law article consists of two parts: (1) the event pattern, which stipulates the behavior that violates the law, and (2) the judgment, which describes the corresponding penalties. In the law article related to Robbery in Figure 1, the event pattern is *Anyone robs public or private property* and the judgment is *be sentenced to imprisonment of not less than three years and not more than ten years*. The event pattern and the corresponding judgment defined by each law article can be viewed as a causal pair: if an event pattern is detected, the corresponding judgment can be inferred from the causal pair. In other words, it is the event information described in the case facts on which the reasoning judgment for the case is based. If we use the fine-grained key event information extracted from the facts to match the event pattern defined in the law articles, the law articles that are applicable to the case could be retrieved accurately and the penalty could be inferred with the judgment in the law article. For example, if we could compress the fact statement in Figure 1 into the fine-grained event in Table 1, we could easily match it with the event pattern defined in Article 263 (see Figure 1). Then the penalty defined in this article can be used as the predicted judgment.

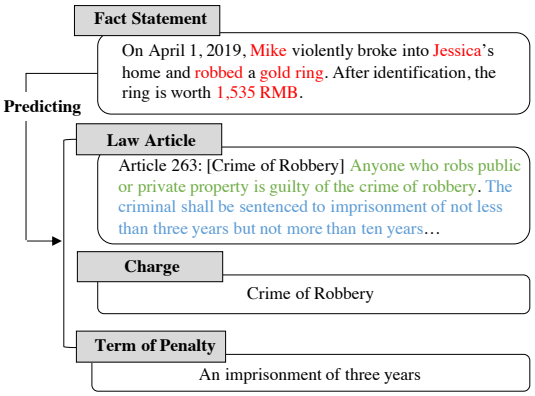

Figure 1: An illustration of legal judgment prediction. The green and blue texts of the law article describe the event pattern and the judicial consequence respectively. The red words are fine-grained event information.

Inspired by this observation, we seek to leverage event information for improving LJP, specifically

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 1: Long Papers, pages 648 - 664

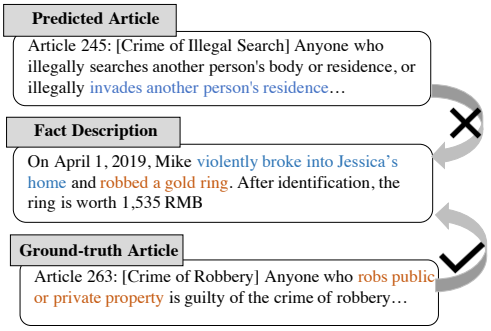

Figure 2: An error example of SOTA models.

	Argument	Role
Who is the criminal?	Mike	Criminal
Who is the victim	Jessica	Victim
What happened?	robbed	Trigger-Rob
What were robbed?	gold ring	Property
What is the price of swag?	1,535 RMB	Quantity
Judgment Results: Article 263, Robbery, three-year imprisonment		

Table 1: An example of the judging process of a real case based on event information.

by (1) extracting the fine-grained key event of the case and then (2) predict the judgment based on the extracted event information (instead of the whole fact statement). To this end, we propose a hierarchical event definition referring to the hierarchy of law articles. Since there is no public LJP dataset that is annotated with event information, we manually annotate a legal event dataset on the top of CAIL (a public LJP dataset widely used by SOTA methods) (Xiao et al., 2018). Nevertheless, event extraction is challenging. So, to guide the learning process, we design output constraints on event extraction (e.g., what role types are compulsory for a given trigger type) and employ them in our model.

Another weakness associated with SOTA methods concerns their failure to exploit the *consistency* constraints among the three LJP subtasks. Specifically, each law article imposes constraints on what charge and term penalty are possible. However, SOTA methods typically frame LJP as a multi-task learning problem in which the three tasks are jointly learned in a model via a shared representation, without guaranteeing that the aforementioned cross-task constraints are satisfied. To address this problem, we introduce consistency constraints.

In sum, our contributions are three-fold. First, we present the first study on leveraging event extraction from case facts to solve LJP tasks. Second, we define a hierarchical event structure for legal cases and collect a new LJP dataset with event annotations. Finally, we propose a model that learns LJP and event extraction jointly subject to two kinds of constraints. Experiments show that our model surpasses the existing SOTA models in performance.

2 Related Work

Legal judgment prediction. LJP has been investigated in the context of different jurisdictions, such as China (Luo et al., 2017; Zhong et al., 2018; Yue et al., 2021; Feng et al., 2021), the U.S. (Katz et al., 2017), Europe (Chalkidis et al., 2019a, 2021), French (Sulea et al., 2017b,a), India (Malik et al., 2021; Paul et al., 2020). While early works relied on rule-based approaches (Kort, 1957; Segal, 1984; Nagel, 1963), later approaches use classification techniques (Aletras et al., 2016; Liu et al., 2015; Sulea et al., 2017a,b; Katz et al., 2017). More recently, neural models are learned to predict judgment results jointly by sharing parameters in a unified framework (Zhong et al., 2018; Xu et al., 2020; Dong and Niu, 2021; Yang et al., 2019; Feng et al., 2019), applying pre-trained language models (Chalkidis et al., 2020, 2021; Xiao et al., 2021; Niklaus et al., 2021; Zhong et al., 2019), exploiting label-attention mechanisms (Wang et al., 2018, 2019), or injecting legal knowledge (Hu et al., 2018; Gan et al., 2021; Zhong et al., 2020). Unlike our work, these works do not explore the use of case events for LJP. Though existing works exploit dependency between subtasks (Zhong et al., 2018; Yang et al., 2019), they merely utilize the subtasks' prediction results as auxiliary features to influence each other and therefore may still predict inconsistent results. In contrast, our cross-task consistency constraints can guarantee that the predictions are consistent.

Event extraction in legal domain. Some works have defined legal events and built models to automatically extract legal events from fact statements using these definitions (Shen et al., 2020; Li et al., 2019; Chen et al., 2020). However, we cannot use these event-annotated legal datasets for two reasons. First, the legal documents in these datasets do not contain legal judgment predictions, so we cannot use them to jointly extract events and make legal judgment predictions. Second, there is a key difference between our work and previous work in terms of how legal events (i.e., the trigger types and argument roles) are defined: while existing works define legal events solely from the perspective of event extraction, we define legal events so that the trigger types and argument roles are useful for LJP.

Dataset	CAIL-small	CAIL-big
#Training Set Cases	96,540	1,489,932
#Validation Set Cases	12,903	–
#Testing Set Cases	24,848	185,647
#Law Articles	101	127
#Charges	117	140
#Term of Penalty	11	11

Table 2: Statistics on CAIL.

3 Dataset and Task Definition

Dataset. We employ as our dataset CAIL (Xiao et al., 2018), a large-scale publicly available Chinese legal document dataset that has been widely used. In CAIL, each judgment document consists of a fact statement and judgment results (law articles, charges and term of penalty). We follow prior works (Xu et al., 2020; Yang et al., 2019) for preprocessing CAIL (see Appendix H). CAIL is composed of two subdatasets: CAIL-big and CAIL-small, and their statistics are shown in Table 2. LJP on CAIL is by no means trivial: there are 127, 140 and 11 categories for article, charge and penalty respectively on CAIL-big.

Task definition. Given a fact statement, LJP on CAIL involves three prediction subtasks $t_a, t_c, t_p \in \mathcal{T}$, which correspond to law article, charge and term of penalty respectively. Following previous works (Xu et al., 2020; Yang et al., 2019), we formalize each subtask $t \in \mathcal{T}$ as a multi-class classification problem and predict for each t the corresponding result $y^t \in \mathcal{Y}^t$, where $\mathcal{Y}^t$ is the label set of t.

4 Baseline LJP Model

We begin by designing a multi-task legal judgment prediction model, which we will use as a baseline and augment with event extraction and constraints in subsequent sections. The framework of our model is shown in Figure 3.

Token representation layer. Given a fact statement represented as a character sequence $D = \{x_1, x_2, ...x_{l_f}\}$, we first encode each character by passing them into a pretrained legal BERT encoder (Zhong et al., 2019).

$$\mathbf{h}_1, \mathbf{h}_2, ...\mathbf{h}_{l_f} = \text{Legal-BERT}(x_1, x_2, ...x_{l_f}) \quad (1)$$

where $\mathbf{H}_f = \{\mathbf{h}_1, \mathbf{h}_2, ...\mathbf{h}_{l_f}\}$ is the hidden vector sequence of the fact statement and l_f is the length of the fact statement.

Generating context features. Next, we generate the context representation of the fact statement by

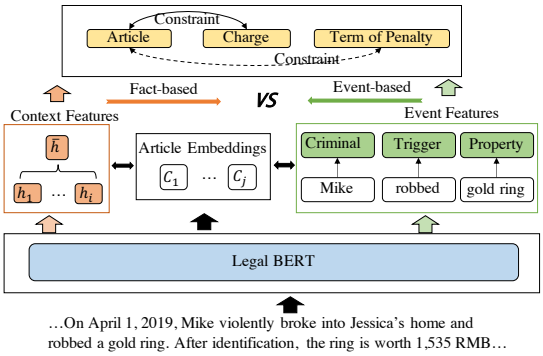

Figure 3: Model architecture. Note that the green box, which computes event features, and the "Constraint"s in the uppermost box are not part of the baseline.

applying a max-pooling layer to $\mathbf{H}_f$:

$$\bar{\mathbf{h}} = maxpooling(\mathbf{h}_1, \mathbf{h}_2, ...\mathbf{h}_{l_f}) \quad (2)$$

Incorporating law article semantics. Using the aforementioned context representation to predict judgment essentially treats each law article as an atomic label, leaving its semantic information unexploited. Inspired by previous work (Chalkidis et al., 2019b; Rios and Kavuluru, 2018), we employ an attention mechanism to incorporate article semantics into the model. Specifically, we match $\bar{\mathbf{h}}$ with all candidate law articles. To do so, we first use the same encoder to encode the character sequence of each law article and obtain the hidden vector sequence $\mathbf{H}_a = \{\mathbf{h}_1, \mathbf{h}_2, ...\mathbf{h}_{l_a}\}$, where l_a is the length of a law article text. Then we apply a max-pooling layer to $\mathbf{H}_a$ to get the context representation $\mathbf{c}$. Next, we use the context representation of the fact statement, $\bar{\mathbf{h}}$, to query all candidate articles in order to mine the most relevant semantics in the article texts. Specifically, we first obtain the relevance scores between $\bar{\mathbf{h}}$ and the j-th article $\mathbf{c}_j$:

$$\alpha_j = \bar{\mathbf{h}}^T \mathbf{W}_c \mathbf{c}_j \quad (3)$$

where $\mathbf{W}_c$ is a trainable matrix. Then the most relevant semantics are summed in a weighted fashion to represent the features from the article texts:

$$\bar{\mathbf{c}} = \sum \frac{exp(\alpha_j)}{\sum_{k=1} exp(\alpha_k)} \mathbf{c}_j \quad (4)$$

where $\bar{\mathbf{c}}$ contains the integrated article semantics.

Legal judgment prediction layer. To predict legal judgment, we input $\bar{\mathbf{h}}$ and $\bar{\mathbf{c}}$ into three task-specific classifiers as follows:

$$\hat{\mathbf{y}}^t = softmax(\mathbf{W}_t[\bar{\mathbf{h}}; \bar{\mathbf{c}}] + \mathbf{b}_t) \quad (5)$$

where $\mathbf{W}_t$ and $\mathbf{b}_t$ are the learnable parameters and $\hat{\mathbf{y}}^t$ is the prediction distribution of task t.

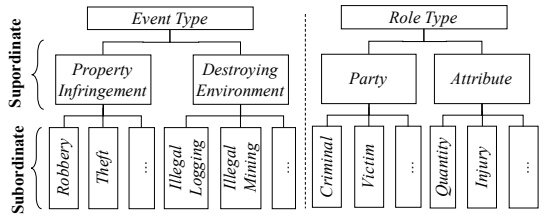

Figure 4: Examples of hierarchical events and roles.

Training. For each legal judgment prediction task t, we employ cross-entropy as the loss function to measure the distance between the predicted $\hat{\mathbf{y}}^t$ and the ground-truth $\mathbf{y}^t$.

$$\mathcal{L}_t = -\sum \mathbf{y}^t log \hat{\mathbf{y}}^t \tag{6}$$

The final loss is composed of the losses of three subtasks, and is defined as follows:

$$\mathcal{L}(\Theta) = \lambda_{t_a}\mathcal{L}_{t_a} + \lambda_{t_c}\mathcal{L}_{t_c} + \lambda_{t_p}\mathcal{L}_{t_p} \tag{7}$$

where hyperparameters λ determine the trade-off between all subtask losses. The model is trained to minimize $\mathcal{L}(\Theta)$.

5 Improving LJP via Event Extraction

In this section, we propose a novel method to leverage event extraction to improve LJP.

5.1 Hierarchical Legal Event Definition

Event definition. Each law article stipulates what event violates this article, so it is easy to define legal events based on law articles. The Chinese law articles have been organized in a hierarchical manner. For example, robbery-related and theft-related articles belong to *Property Infringement*, which is the general name of robbery-related and theft-related articles. We define legal events following this hierarchy. As shown in Figure 4, *Property Infringement* is treated as a superordinate event type, whereas *Robbery* and *Theft* are treated as subordinate event types. This hierarchy can express the connections between different legal events.

Trigger and role definitions. An event trigger is a word that realizes the occurrence of an event and has a type. There is a one-to-one correspondence between event type and trigger type. For example, the *Robbery* event has the trigger type *Trigger-Rob*.

Next, we define the roles for each event such that they reflect the key elements of the event that would be useful for making legal judgments. For example, the *Criminal* and *Victim* roles specify the parties involved in a case, whereas the *Quantity* role measures the value of loot, based on which term penalty is derived. We define the roles in a hierarchical manner. As seen in Figure 4, the *Party* arguments are the people involved in the cases, and its subordinate roles include *Criminal* and *Victim*.[1]

5.2 Dataset Collection

To investigate the use of event extraction for LJP, we manually create an event-annotated LJP dataset since no such dataset is publicly available.

Step 1: judgment document collection. We construct LJP-E, our event-annotated dataset, based on CAIL. Specifically, we first analyze the performance of the SOTA models (Xu et al., 2020; Zhong et al., 2018) on the validation portion of CAIL-small and identify the 15 law articles for which they achieved poor performance, and then select a subset of the cases that can be judged by these 15 law articles for annotation. This subset consists of 1367 documents (957 as training set, 136 as validation set and 274 as test set). We henceforth refer to this set of judgment documents as D_o.

Step 2: event trigger and argument role annotation. Next, we hire two annotators to manually produce event triggers and argument roles for each case in D_o after giving them a three-hour tutorial on how to annotate events. The annotators are native speakers of Chinese who are graduate students in NLP with significant experience with working on legal problems (none of them are the authors).

The annotation process. Given the fact statement and the gold law article of a case, each annotator is asked to independently highlight the salient words in the fact statement that reflect the core event of the case and correlate well with the event pattern of the law article. Then each of them is asked to (1) select a trigger word and assign it a subordinate trigger type, and (2) assign a subordinate role type to each of its arguments from a predefined role list. The trigger type and role type inventories were designed by the authors after having read a large number of fact statements and corresponding articles. Inter-annotator agreement numbers can be found in Appendix E.

After the above steps, each case in D_o is annotated with a trigger, its type, its arguments and roles. The average number of arguments per event is 4.13.

[1]Details of the event and role definitions together with their explanations can be found in Appendix A and B.

There are 16 distinct subordinate roles and 15 distinct subordinate trigger types.[2] Each disagreement between the annotators is resolved via discussion.

5.3 Hierarchical Event Extraction

To make use of the event annotations, we augment our baseline model with a hierarchical event extraction layer that detects event triggers and arguments and determines trigger types and arguments roles (see Figure 3). The resulting model simultaneously learns event extraction and LJP.

We formalize event extraction as a token labeling problem. Given the hidden vectors of the fact tokens $\mathbf{H}_{l_f} = \{\mathbf{h}_1, \mathbf{h}_2, ...\mathbf{h}_{l_f}\}$, we assign each token a subordinate trigger (if it is part of a trigger) or a subordinate role type (if it is part of an argument).

The hierarchical event extraction layer consists of two modules: (1) a *superordinate* module that attends each hidden vector to all superordinate types/roles for obtaining their correlations, and (2) a *subordinate* module that computes the subordinate type/role probability distribution based on hierarchical information, as described below.

For a specific superordinate type/role j, we represent its semantic features with a trainable vector $\mathbf{p}_j$. We adopt a fully-connected layer to calculate the correlation score between hidden vector $\mathbf{h}_i$ and superordinate type/role $\mathbf{p}_j$.

$$u_{ij} = \mathbf{W}_p[\mathbf{h}_i; \mathbf{p}_j] \tag{8}$$

where u_{ij} represents the correlation score and $[;]$ denotes the concatenation of two vectors. Then, we apply a softmax operation to get the superordinate type/role feature for each token x_i.

$$\beta_{ij} = \frac{exp(u_{ij})}{\sum_{k=1} exp(u_{ik})} \tag{9}$$

$$\mathbf{o}_i = \sum_{j=1} \beta_{ij} \mathbf{p}_j \tag{10}$$

where $\mathbf{o}_i$ is the integrated superordinate type/role feature, which provides superordinate-oriented information useful for predicting subordinate types/roles. Next, we concatenate each $\mathbf{h}_i$ with $\mathbf{o}_i$ as the input feature for the trigger type and argument role classifier and estimate the probability that token x_i belongs to subordinate type/role r_j as follows:

$$s_r(x_i, \hat{y}^{r_j}_{(i)}) = \frac{exp(\mathbf{q}_j^T[\mathbf{h}_i; \mathbf{o}_i])}{\sum_{k=1} exp(\mathbf{q}_k^T[\mathbf{h}_i; \mathbf{o}_i]]} \tag{11}$$

[2]Statistics of LJP-E can be found in Appendix D.

where $\mathbf{q}_j$ is the trainable vector of r_j. After obtaining the type/role probability distribution of x_i, we apply a CRF (Lafferty et al., 2001) to produce the sequence of types/roles with the highest score, where the score of a sequence of types/roles $\hat{\mathbf{y}}^r = \{\hat{y}^r_{(1)}, \hat{y}^r_{(2)}...\}$ is computed as:

$$score(D, \hat{\mathbf{y}}^r) = \sum_{i=1}^{l_f} \mathbf{T}_{\hat{y}^r_{(i-1)}, \hat{y}^r_{(i)}} + \sum_{i=1}^{l_f} s_r(x_i, \hat{y}^r_{(i)}) \tag{12}$$

Here, $\mathbf{T}$ is the score of transitioning from one tag to another tag.

Instead of predicting LJP based on the context fact representation $\bar{\mathbf{h}}$, we replace it with the detected event features. Specifically, we input the extracted trigger word and arguments with their type/role embeddings into the three task-specific classifiers. Denote an extracted span as $\mathbf{H}_s = \{\mathbf{h}_1, \mathbf{h}_2, ...\mathbf{h}_{l_s}\}$. We apply a max-pooling layer to each span and concatenate with the corresponding subordinate type/role embedding $\mathbf{q}$:

$$\mathbf{g}_i = [maxpooling(\mathbf{H}_s); \mathbf{q}] \tag{13}$$

where $\mathbf{g}_i$ denotes the representation of span i, which contains both semantics and subordinate type/role features. Based on $\mathbf{g}_i$, we can calculate the context span representation $\bar{\mathbf{g}}$ as follows:

$$\bar{\mathbf{g}} = maxpooling(\mathbf{g}_1, \mathbf{g}_2...) \tag{14}$$

which is used to replace the context fact representation $\bar{\mathbf{h}}$ in Equation 2.

Training. The event extraction loss is defined as:

$$\mathcal{L}_r = -\sum log \frac{e^{score(D, \mathbf{y}^r)}}{\sum e^{score(D, \hat{\mathbf{y}}^r)}} \tag{15}$$

where $\mathbf{y}^r$ is the gold tag sequence. We incorporate $\mathcal{L}_r$ into he total loss in Equation 7 as follows:

$$\mathcal{L}(\Theta) = \lambda_{t_a}\mathcal{L}_{t_a} + \lambda_{t_c}\mathcal{L}_{t_c} + \lambda_{t_p}\mathcal{L}_{t_p} + \lambda_r\mathcal{L}_r \tag{16}$$

6 Exploiting Constraints

To improve model performance, we explore two types of constraints, as described below.

6.1 Event-Based Constraints

Event-based constraints are output constraints on events. We propose two such constraints.

Absolute constraint. For a legal event, the trigger must appear exactly once and certain roles are compulsory (e.g., subordinate role *Criminal* should appear at least once). If the trigger is missing, we impose the following penalty:

$$\mathcal{P}_t = \sum_{\tilde{r}\in\mathcal{TG}} \sum_{i=1}^{l_f} s_r(x_i, \hat{y}^{\tilde{r}}_{(i)}) - \max_{i,\tilde{r}\in\mathcal{TG}}[s_r(x_i, \hat{y}^{\tilde{r}}_{(i)})] + |1 - \max_{i,\tilde{r}\in\mathcal{TG}}[s_r(x_i, \hat{y}^{\tilde{r}}_{(i)})]| \tag{17}$$

where $\mathcal{TG}$ is the trigger inventory.[3] If a required role r is missing, we impose the following penalty:

$$\mathcal{P}_r = |1 - \max_i[s_r(x_i, \hat{y}^{r}_{(i)})]| \tag{18}$$

Event-Based consistency constraint. If a trigger type is detected, all and only its related roles should be detected. For example, if a *Illegal Doctoring* event is detected, no roles related to *Illegal Logging* should be predicted. If a trigger r is predicted, we impose the following penalty:

$$\mathcal{P}_e = \sum_{\tilde{r}\in\mathcal{R}^+} |1 - \max_i[s_r(x_i, \hat{y}^{\tilde{r}}_{(i)})]| + \sum_{\tilde{r}\in\mathcal{R}^-} \sum_{i=1}^{l_f} s_r(x_i, \hat{y}^{\tilde{r}}_{(i)}) \tag{19}$$

where $\mathcal{R}^+$ is the set of roles that should occur if r is predicted, and $\mathcal{R}^-$ is the set of roles that cannot occur. We sum all the penalty terms and incorporate them into the total loss as follows:

$$\mathcal{L}(\Theta) = \lambda_{t_a}\mathcal{L}_{t_a} + \lambda_{t_c}\mathcal{L}_{t_c} + \lambda_{t_p}\mathcal{L}_{t_p} + \lambda_r\mathcal{L}_r + \lambda_p \sum \mathcal{P}_i \tag{20}$$

6.2 Cross-Task Consistency Constraints

While the multi-task learning setup employed by our model allows subtasks of LJP to benefit each other via the shared representation layer, it fails to exploit the dependency explicitly that exist among them. Below we exploit two such dependencies, one between law article and charge and the other between law article and term of penalty.

Each law article states the allowable charges and range of term of penalty. Hence, we can utilize these dependencies to constrain (and hopefully improve) the prediction of charge and term of penalty using the predicted law article. More specifically, we make the model learn how to predict charge and term of penalty based on the predicted article during training by modifying the cross entropy loss as follows. If the law article is predicted correctly by the model, then when calculating $\mathcal{L}_{t_c}$ (i.e., the cross-entropy loss associated with the charge prediction task), we mask each term in the loss corresponding to a charge that is not allowed according to the predicted article:

$$\mathcal{L}_{t_c} = -\sum\sum mask * y^{t_c} log\hat{y}^{t_c} \tag{21}$$

where $mask$ is equal to 0 if the charge is not allowed according to the predicted article and 1 otherwise. However, if the article is predicted incorrectly by the model, $\mathcal{L}_{t_c}$ is the standard cross entropy loss. Intuitively, through masking, the model is forced to predict a charge that is allowed according to the predicted article. During testing, since we do not know whether the law article is predicted correctly or not, we always mask the charge probability distribution according to the predicted article. We adopt the same strategy to compute $\mathcal{L}_{t_p}$ when enforcing the consistency constraint between law article prediction and term of penalty prediction.

7 Evaluation

7.1 Experimental Setup

We train our model *EPM* using the pre-training and fine-tuning strategy. Specifically, we pre-train *EPM* without event components on the training portion of CAIL (Table 2), and then fine-tune *EPM* on the training portion of LJP-E, our event-annotated LJP dataset, to learn from the event annotations.

As for the encoder, the maximum fact length is set to 512. For training, we utilize the Adam optimizer with learning rate of 10^{-4} and the batch size is 32. The warmup step is 3000. For the hyperparameters, λ, in the loss function, the best setting is {0.5, 0.5, 0.4, 0.2, 0.1} for $\{\lambda_{t_a},\lambda_{t_c},\lambda_{t_p},\lambda_r,\lambda_p\}$. Models are trained for a maximum of 20 epochs. [4] LJP results are reported in terms of Accuracy (Acc), Macro-Precision (MP), Macro-Recall (MR) and Macro-F1 (F1).

7.2 Comparison with the SOTA

We compare *EPM* with SOTA models on the test portion of our annotated dataset LJP-E in Table 3. and the official test portion of the CAIL dataset in

[3] An explanation of the penalty functions in Equation 17, 18 and 19 can be found in Appendix K

[4] Details of experimental setup can be found in Appendix I

		Law Article				Charge				Term of Penalty			
		Acc%	**MP%**	**MR%**	**F1%**	**Acc%**	**MP%**	**MR%**	**F1%**	**Acc%**	**MP%**	**MR%**	**F1%**
1	MLAC	83.75	71.49	71.79	70.05	73.20	52.82	55.93	52.19	23.57	17.92	17.22	16.38
2	TOPJUDGE	86.46	75.51	75.07	73.97	75.16	56.04	58.96	55.34	23.82	18.59	18.43	17.63
3	MBPFN	86.72	86.72	75.60	74.28	73.95	73.95	56.48	54.00	27.53	27.53	17.97	19.65
4	LADAN	89.92	78.13	78.01	77.06	79.12	58.54	61.87	58.35	26.06	20.86	18.03	16.58
5	NeurJudge	87.87	81.17	82.68	80.41	76.04	61.95	60.07	59.46	27.88	20.99	16.81	18.51
6	EPM	93.85	91.15	89.14	89.37	79.37	61.14	63.15	60.79	28.51	28.27	23.58	23.23
7	w/ gold	97.05	95.63	93.42	93.82	86.98	70.45	73.08	70.92	33.16	30.28	23.52	24.11
8	TOPJUDGE+Event	88.84	78.03	79.72	77.39	77.24	60.70	60.57	57.40	27.49	22.38	18.17	18.10

Table 3: Comparisons with the SOTA models on LJP-E.

		Law Article				Charge				Term of Penalty			
		Acc%	**MP%**	**MR%**	**F1%**	**Acc%**	**MP%**	**MR%**	**F1%**	**Acc%**	**MP%**	**MR%**	**F1%**
1	MLAC	94.90	79.06	66.91	69.41	94.72	83.42	72.38	75.62	56.43	46.87	40.43	41.89
2	TOPJUDGE	95.83	82.10	71.94	74.32	95.77	85.95	77.11	79.58	58.09	47.73	42.47	44.07
3	MBPFN	95.67	84.00	74.40	76.44	94.37	85.60	75.86	77.98	55.48	47.27	38.26	40.01
4	LADAN	95.78	84.93	75.88	78.79	94.58	85.52	77.36	80.04	56.34	47.76	40.48	42.02
5	NeurJudge	95.59	84.01	75.54	77.06	94.12	85.48	77.21	79.83	55.52	47.25	40.76	42.03
6	EPM	96.63	85.93	77.60	79.72	95.88	88.67	79.49	81.99	58.19	51.50	43.25	44.99
7	EPM@G	96.72	85.79	79.68	81.77	96.45	88.78	81.93	82.84	58.67	53.93	45.86	46.58
8	MLAC+EPM	95.50	79.71	70.29	72.81	95.45	84.18	73.14	75.86	57.39	47.08	41.53	43.07
9	TOPJUDGE+EPM	96.01	83.68	74.77	77.26	95.86	86.21	78.67	81.23	58.11	48.20	44.30	45.07
10	MPBFN+EPM	95.81	83.36	74.61	76.39	95.62	86.34	77.34	79.35	57.53	50.04	40.46	42.01
11	LADAN+EPM	96.15	84.90	76.54	79.26	95.96	88.07	78.98	81.79	58.40	50.36	42.71	44.17
12	NeurJudge+EPM	96.20	85.16	77.83	78.21	94.77	89.75	77.46	80.19	57.81	49.36	41.77	43.79
13	TOPJUDGE+Event	95.93	83.55	73.03	75.86	95.82	86.34	77.20	80.29	58.21	47.73	44.36	45.00

Table 4: Comparisons with the SOTA models on CAIL-big.

		Law Article				Charge				Term of Penalty			
		Acc%	**MP%**	**MR%**	**F1%**	**Acc%**	**MP%**	**MR%**	**F1%**	**Acc%**	**MP%**	**MR%**	**F1%**
1	MLAC	73.02	69.27	66.14	64.23	74.73	72.65	69.56	68.36	36.45	34.50	29.95	29.64
2	TOPJUDGE	78.60	76.59	74.84	73.72	81.17	81.87	80.57	79.96	35.70	32.81	31.03	31.49
3	MPBFN	76.83	74.57	71.45	70.57	80.17	78.88	75.65	75.68	36.18	33.67	30.08	29.43
4	LADAN	78.70	74.95	75.61	73.83	82.86	81.69	80.40	80.05	36.14	31.85	29.67	29.28
5	NeuralJudge	79.02	75.69	75.23	74.87	81.22	77.51	78.17	77.99	36.84	34.80	32.22	32.48
6	EPM	84.65	80.82	77.55	78.10	84.10	84.55	80.22	81.43	36.69	35.60	32.70	32.99
7	EPM@G	85.65	83.51	78.56	79.76	85.39	85.54	80.74	82.16	37.59	35.32	32.51	33.14
8	MLAC+EPM	81.16	79.29	71.17	72.08	81.13	80.88	75.60	76.12	36.04	30.92	30.69	29.77
9	Topjudge+EPM	83.73	80.29	76.88	77.42	83.67	83.72	80.06	80.87	36.41	33.02	31.88	31.55
10	MPBFN+EPM	82.33	76.33	75.26	74.56	81.82	79.81	77.37	77.51	36.40	34.01	31.41	32.31
11	LADAN+EPM	83.59	78.65	77.99	77.10	84.91	83.17	81.31	81.54	36.54	34.06	31.14	32.06
12	NeurJudge+EPM	84.01	77.43	77.11	76.83	83.12	78.13	78.24	78.15	37.01	35.24	32.91	32.51
13	TOPJUDGE+Event	80.56	77.67	75.67	75.28	82.79	82.52	79.43	80.01	36.66	33.34	31.69	31.53

Table 5: Comparisons with the SOTA models on CAIL-small.

Table 4 and 5. Since LJP-E only contains the 15 case types of CAIL, when applying *EPM* on the CAIL test set we use the pretrained version of *EPM* (i.e., without fine-tuning) to predict samples that do not belong to the 15 types and use the fine-tuned version of *EPM* to predict samples that belong to one of the 15 types. In order to determine whether a sample belongs to one of the 15 types, we train a binary classifier using legal BERT on the training set of CAIL. We refer to this model as the *Switch*.[5]

We compare *EPM* with four SOTA neural models: (1) **MLAC** (Luo et al., 2017), which jointly modeled charge prediction and the relevant article extraction task in a unified framework. Here, we add a fully-connected layer in order to predict the term of penalty; (2) **TOPJUDGE** (Zhong et al., 2018), which formalized the subtasks of LJP in a joint framework as a directed acyclic graph in which the subtasks share parameters; (3) **MPBFN** (Yang et al., 2019), which proposed a multi-perspective forward and backward prediction framework to make the sharing of parameters by different subtasks effectively, as well as a number embedding method for term of penalty prediction; (4) **LADAN** (Xu et al., 2020), which developed a

[5]Details of the Switch can be found in Appendix J.

graph networks to learn the subtle differences between law articles in order to extract compelling discriminative features from fact statements; and (5) **NeurJudge** (Yue et al., 2021), which utilized the results of intermediate subtasks to separate the fact statement into different circumstances and exploits them to make the predictions of other subtasks.

As shown in Tables 3, 4 and 5, *EPM* (row 6) achieves the best results, substantially outperforming not only MLAC but also TOPJUDGE, MPBFN, LADAN and NeurJudge, which *further leverage extensions like number embedding and graph networks*, particularly on law article prediction.

Next, we conduct two *oracle* experiments involving *EPM*. First, we use *gold* rather than *predicted* event annotations to make predictions for the three subtasks.[6] The results, which are show in row 7 of Table 3, show that considerably better results can be obtained when gold event annotations are used. These results suggest that existing LJP results can be substantially improved by improving event extraction. Next, we assume that the *Switch* is perfect when obtaining the *EPM* results on CAIL. Perhaps not surprisingly, results, which are shown in row 7 of Table 4 and 5, are better w.r.t. all subtasks.

Further, we apply *EPM* to *MLAC*, *TOPJUDGE*, *MPBFN*, *LADAN* and *NeurJudge* on CAIL, the five SOTA models following the same scheme (i.e., use fine-tuned *EPM* to classify when the *Switch* says the sample belongs to the 15 types and use the SOTA model to classify otherwise), showing the results in rows 8 to 12 in Table 4 and 5. We see that *EPM* can also improve the performance of the four SOTA models, yielding new SOTA results.

Finally, we examine whether *modifying* a SOTA model, *TOPJUDGE*, by having it jointly perform event extraction and the LJP tasks can improve its performance. To do so, we replace its CNN encoder by an LSTM and feed the LJP classifiers with the extracted events rather than the case facts in the same way as in *EPM*. We can see that *TOPJUDGE+Event* outperforms *TOPJUDGE*, which shows the usefulness of event information. However, *TOPJUDGE+Event* underperforms *TOPJUDGE+EPM*. This suggests that better LJP results can be achieved by treating *TOPJUDGE* as a black box (by exploiting event information using the *Switch*) rather than a glass box (by modifying the model to learn from event annotations).

[6]Event extraction results in *EPM* are as follows: 53.75% (R), 47.52% (P), and 50.37% (F1) for trigger detection and 55.69% (R), 49.88% (P), and 52.59% (F1) for role prediction.

7.3 Usefulness of Events and Constraints

We conduct experiments on the ablated versions of *EPM*. Ablation results on *LJP-E* and *CAIL* are shown in Tables 6 and 7, 8.

Event extraction. To test the usefulness of event extraction, we delete all event components from *EPM*. Results are shown in row 2. As we can see, performance degrades substantially on all three subtasks in terms of both Acc and F1.

Event-based constraints. Next, we evaluate the usefulness of the two event-based constraints (Section 6.1) on the outputs of event extraction. Removing the absolute constraint (*w/o CSTR1*, row 3) or the event-based consistency constraint (*w/o CSTR2*, row 4) generally yields worse results in terms of both Acc and F1. In particular, removing the consistency constraint generally provides bigger deterioration than the absolute constraint.

Cross-task consistency constraints. We also evaluate the cross-task consistency constraints. Removing the article-charge constraint (*w/o DEP1*, row 5) or the article-term constraint (*w/o DEP2*, row 6) negatively impacts performance, with the largest negative impact observed on charge prediction. While these constraints are intended to employ the predicted law article results to improve charge prediction and term prediction, we see that law article performance also deteriorates.

Superordinate types. So far, we have assumed that hierarchical event extraction would be beneficial to LJP. To better understand whether the hierarchy is indeed useful, we evaluate a version of *EPM* *without* using superordinate features. In other words, the model predicts the subordinate types/roles directly. Results are shown in row 7. Comparing rows 1 and 7, we see that Acc and F1 scores drop across all subtasks when superordinate features are not used, indicating their usefulness.

Event extraction as an auxiliary task. In *EPM*, we use the predicted event features as inputs for the three LJP task classifiers. Another way to exploit event information would be to treat event extraction as an auxiliary task in the model by having it share encoders with the LJP tasks. Results of treating event extraction as an auxiliary task are shown in row 8. As we can see, these results are worse than those of *EPM* (row 1), which means that *EPM*'s way of exploiting event information is better, but they are better than those when event information is

		Law Article				Charge				Term of Penalty			
		Acc%	**MP%**	**MR%**	**F1%**	**Acc%**	**MP%**	**MR%**	**F1%**	**Acc%**	**MP%**	**MR%**	**F1%**
1	EPM	93.85	91.15	89.14	89.37	79.37	61.14	63.15	60.79	28.51	28.27	23.58	23.23
2	w/o event	86.17	76.49	75.66	75.19	73.21	56.94	56.46	55.46	26.25	18.78	15.71	15.06
3	w/o CSTR1	88.44	83.22	80.77	80.00	74.21	57.06	59.37	56.54	27.85	18.01	18.16	16.87
4	w/o CSTR2	86.96	77.95	76.17	76.84	74.69	57.29	56.98	55.99	27.77	19.18	17.55	16.99
5	w/o DEP1	91.15	86.29	85.23	84.96	73.62	56.84	58.69	56.71	23.35	16.29	16.06	15.23
6	w/o DEP2	91.89	90.98	87.41	88.30	78.38	60.48	61.43	59.31	23.31	17.30	14.12	14.49
7	w/o hierarchy	87.21	79.20	76.52	76.38	73.95	58.23	59.50	57.36	23.59	19.42	16.77	16.32
8	w/ auxiliary	92.62	85.24	83.95	83.90	76.18	55.08	59.05	56.39	25.31	21.06	16.58	15.88

Table 6: Ablation results on LJP-E.

		Law Article				Charge				Term of Penalty			
		Acc%	**MP%**	**MR%**	**F1%**	**Acc%**	**MP%**	**MR%**	**F1%**	**Acc%**	**MP%**	**MR%**	**F1%**
1	EPM	96.63	85.93	77.60	79.72	95.88	88.67	79.49	81.99	58.19	51.50	43.25	44.99
2	w/o event	95.32	84.25	72.47	75.40	93.23	81.90	74.74	76.24	55.78	44.29	41.52	42.52
3	w/o CSTR1	95.41	85.07	73.54	76.37	94.20	85.66	77.81	78.99	56.97	47.21	41.04	43.39
4	w/o CSTR2	95.38	84.85	73.29	75.82	94.26	85.24	77.45	78.51	57.36	46.54	40.63	42.87
5	w/o DEP1	95.10	85.30	73.34	76.10	94.91	83.53	75.18	77.41	56.68	46.64	40.63	43.07
6	w/o DEP2	95.29	85.07	73.48	76.06	94.51	85.58	77.44	78.78	55.63	43.24	41.38	42.81
7	w/o hierarchy	95.43	85.32	73.90	76.81	94.24	85.78	78.11	79.24	56.87	47.06	40.89	43.21
8	w/ auxiliary	96.35	86.21	73.81	76.93	95.58	87.54	76.39	79.53	57.12	48.58	42.26	42.98

Table 7: Ablation results on CAIL-big.

		Law Article				Charge				Term of Penalty			
		Acc%	**MP%**	**MR%**	**F1%**	**Acc%**	**MP%**	**MR%**	**F1%**	**Acc%**	**MP%**	**MR%**	**F1%**
1	EPM	84.65	80.82	77.55	78.10	84.10	84.55	80.22	81.43	36.69	35.60	32.70	32.99
2	w/o event	77.73	78.89	75.22	74.49	81.85	82.09	76.48	78.76	33.67	31.73	28.33	28.88
3	w/o CSTR1	83.91	81.57	75.55	76.60	83.36	84.15	77.93	79.43	35.47	30.86	29.67	29.50
4	w/o CSTR2	83.62	81.73	75.56	76.65	83.80	84.70	78.62	80.01	35.17	32.40	29.14	29.45
5	w/o DEP1	82.72	81.33	75.42	76.45	81.20	80.51	76.42	76.67	34.16	32.20	27.88	28.30
6	w/o DEP2	83.40	81.30	75.24	76.33	83.08	84.41	78.06	79.47	32.57	30.07	28.38	28.26
7	w/o hierarchy	83.79	81.50	75.65	76.68	83.37	84.23	79.14	80.13	34.67	31.46	28.95	29.17
8	w/ auxiliary	83.99	80.70	76.22	76.55	84.51	84.60	80.76	81.35	34.29	32.06	29.58	29.22

Table 8: Ablation on CAIL-small.

not use (row 2), which means that using predicted events for LJP is still better than not using them.

7.4 Qualitative Analysis

Next, we perform a qualitative analysis of *EPM* to better understand the role played by event information and constraints. In CAIL, the data distribution of term penalty for the same law article is skewed towards larger penalty values, thus causing *EPM* to inherit this bias in its prediction of term penalty when cross-task consistency constraints are not used. However, when constraints are used, *EPM* was forced to only predict those term penalties that are allowed by the predicted law article and was thus more robust to the skewed data distribution. As for events, the use of event information prevents *EPM* from focusing on certain words in a case fact that could trigger the prediction of wrong law articles. A detailed analysis can be found in Appendix F.

8 Conclusion

We proposed the first model that uses event extraction and hand-crafted constraints to improve LJP, achieving SOTA results. To facilitate future research, we make our codes and annotations publicly available at https://github.com/WAPAY/EPM.

Acknowledgments

We thank the three anonymous reviewers for their comments on an earlier draft of this paper. This work was supported in part by the National Natural Science Foundation of China (No. 61802167), the US National Science Foundation (Grant IIS1528037). Chuanyi Li is the corresponding author. Any opinions, findings, conclusions or recommendations expressed in this paper are those of the authors and do not necessarily reflect the views or official policies, either expressed or implied, of the funding agencies.

Ethics Statement

Automatic legal judgment prediction is a sensitive research area. The proposed EPM model is a preliminary work and is not ready to be productized. The goal of designing the EPM model is to surpass the performances of existing SOAT approaches which are not ready to be productized either.

Legal cases contain personal privacy information. Therefore, we use a public dataset that has been anonymized, i.e., CAIL, which is collected from the official website for disclosing legal case information. The private information of the cases has been anonymized when the cases are published on the official website.

References

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.

Jun Araki, Lamana Mulaffer, Arun Pandian, Yukari Yamakawa, Kemal Oflazer, and Teruko Mitamura. 2018. Interoperable annotation of events and event relations across domains. In *Proceedings 14th Joint ACL-ISO Workshop on Interoperable Semantic Annotation*, pages 10–20.

Reuben Binns. 2019. Human judgment in algorithmic loops: Individual justice and automated decision-making. *Regulation & Governance*.

Ilias Chalkidis, Ion Androutsopoulos, and Nikolaos Aletras. 2019a. Neural legal judgment prediction in english. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019*, pages 4317–4323.

Ilias Chalkidis, Manos Fergadiotis, Prodromos Malakasiotis, Nikolaos Aletras, and Ion Androutsopoulos. 2020. LEGAL-BERT: "preparing the muppets for court'". In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2898–2904.

Ilias Chalkidis, Manos Fergadiotis, Prodromos Malakasiotis, and Ion Androutsopoulos. 2019b. Large-scale multi-label text classification on EU legislation. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019*, pages 6314–6322.

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, NAACL-HLT 2021*, pages 226–241.

Yanguang Chen, Yuanyuan Sun, Zhihao Yang, and Hongfei Lin. 2020. Joint entity and relation extraction for legal documents with legal feature enhancement. In *Proceedings of the 28th International Conference on Computational Linguistics, COLING 2020*, pages 1561–1571.

Qian Dong and Shuzi Niu. 2021. Legal judgment prediction via relational learning. In *SIGIR '21: The 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 983–992.

Yi Feng, Chuanyi Li, Jidong Ge, and Bin Luo. 2019. Improving statute prediction via mining correlations between statutes. In *Proceedings of The 11th Asian Conference on Machine Learning, ACML 2019*, pages 710–725.

Yi Feng, Chuanyi Li, Jidong Ge, Bin Luo, and Vincent Ng. 2021. Recommending statutes: A portable method based on neural networks. *ACM Trans. Knowl. Discov. Data*, 15(2):16:1–16:22.

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*, pages 12866–12874.

Zikun Hu, Xiang Li, Cunchao Tu, Zhiyuan Liu, and Maosong Sun. 2018. Few-shot charge prediction with discriminative legal attributes. In *Proceedings of the 27th International Conference on Computational Linguistics, COLING 2018*, pages 487–498.

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.

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.

John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning (ICML 2001)*, pages 282–289.

Chuanyi Li, Yu Sheng, Jidong Ge, and Bin Luo. 2019. Apply event extraction techniques to the judicial field. In *Proceedings of the 2019 ACM International Joint Conference on Pervasive and Ubiquitous Computing and Proceedings of the 2019 ACM International Symposium on Wearable Computers, UbiComp/ISWC 2019*, pages 492–497.

Yi-Hung Liu, Yen-Liang Chen, and Wu-Liang Ho. 2015. Predicting associated statutes for legal problems. *Inf. Process. Manag.*, 51(1):194–211.

Bingfeng Luo, Yansong Feng, Jianbo Xu, Xiang Zhang, and Dongyan Zhao. 2017. Learning to predict charges for criminal cases with legal basis. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017*, pages 2727–2736.

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, ACL 2021*, pages 4046–4062.

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. *CoRR*, abs/2110.00806.

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, COLING 2020*, pages 1011–1022.

Anthony Rios and Ramakanth Kavuluru. 2018. Few-shot and zero-shot multi-label learning for structured label spaces. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3132–3142.

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

Shirong Shen, Guilin Qi, Zhen Li, Sheng Bi, and Lusheng Wang. 2020. Hierarchical chinese legal event extraction via pedal attention mechanism. In *Proceedings of the 28th International Conference on Computational Linguistics, COLING 2020*, pages 100–113.

Octavia-Maria Sulea, Marcos Zampieri, Shervin Malmasi, Mihaela Vela, Liviu P. Dinu, and Josef van Genabith. 2017a. Exploring the use of text classification in the legal domain. In *Proceedings of the Second Workshop on Automated Semantic Analysis of Information in Legal Texts co-located with the 16th International Conference on Artificial Intelligence and Law (ICAIL 2017)*, volume 2143 of *CEUR Workshop Proceedings*.

Octavia-Maria Sulea, Marcos Zampieri, Mihaela Vela, and Josef van Genabith. 2017b. 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*, pages 716–722.

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, SIGIR 2019*, pages 325–334.

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, SIGIR 2018*, pages 485–494.

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. *CoRR*, abs/1807.02478.

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

Wenmian Yang, Weijia Jia, Xiaojie Zhou, and Yutao Luo. 2019. Legal judgment prediction via multi-perspective bi-feedback network. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019*, pages 4085–4091.

Linan Yue, Qi Liu, Binbin Jin, Han Wu, Kai Zhang, Yanqing An, Mingyue Cheng, Biao Yin, and Dayong Wu. 2021. Neurjudge: A circumstance-aware neural framework for legal judgment prediction. In *SIGIR '21: The 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 973–982.

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.

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

Haoxi Zhong, Zhengyan Zhang, Zhiyuan Liu, and Maosong Sun. 2019. Open chinese language pre-trained model zoo. Technical report.

A The Definition of Hierarchical Event

Table 9 shows the hierarchical relationship between legal events.

Property Infringement: seizing public or private property for the purpose of illegal possession or deliberately destroying public or private property.

- Robbery: robbing public or private property by violence, coercion or other means.
- Theft: stealing public and private property.
- Fraud: swindling public and private property.
- Racketeering: extorting public and private property.

Personal Rights Infringement: the illegal violation of citizens' personal rights and democratic rights.

- Intentional Injury: deliberately and illegally harming the health of others.
- Rape: forcibly having sexual intercourse with the victim by violence, threat or other means against the will of the victim.
- Kidnapping: using violence, coercion or other means to control others, restrict their personal freedom, or coerce others as hostages for the purpose of extorting money or property.

Disturbing Public Order: gathering people to disturb public place and traffic order, and resisting or hindering state security administration personnel from performing their duties according to law.

- Obstructing Official Duties: obstructing State functionaries from performing their duties according to law by means of violence or threat.
- Forgery: illegally manufacturing, altering, buying and selling official documents, certificates and seals of state organs.
- Gambling: gathering people to gamble or gambling for the purpose of profit.

Endangering Public Health: endangering the state's health management.

- Illegal Doctoring: engaging in diagnosis and treatment activities without medical qualification.

Destroying Environment: intentionally violating environmental protection laws, polluting or damaging environmental resources.

- Endangering Rare Wildlife: illegally hunting and killing precious and endangered wild animals under special state protection.
- Illegal Logging: cutting down state, collective or individual owned forests without authorization.

Drug: violation of relevant national and international drug control laws and regulations.

- Drug Possession: illegally possessing drugs.
- Drug Cultivation: cultivating opium poppy, marijuana and other original drug plants.

Superordinate type	Subordinate type
Property Infringement	Robbery
	Theft
	Fraud
	Racketeering
Personal Rights Infringement	Intentional Injury
	Rape
	Kidnapping
Disturbing Public Order	Obstructing Official Duties
	Forgery
	Gambling
Endangering Public Health	Illegal Doctoring
Destroying Environment	Endangering Rare Wildlife
	Illegal Logging
Drug	Drug Possession
	Drug Cultivation

Table 9: Hierarchical event types.

Superordinate type	Subordinate type
Party	Criminal
	Victim
	Officer
State	Qualified
	Intention
	Method
Object	Property
	Instrument
	Animal
	Plant
	Drug
	Drug plant
	Gambling device
	License
Attribute	Quantity Attribute
	Injury Attribute

Table 10: Hierarchical role types.

B The Definition of Hierarchical Role

Table 10 shows the hierarchical relationship between role types.

Party: a person who enters a lawsuit.

- Criminal: the person violating law articles.
- Victim: the aggrieved party.
- Officer: government officials.

State: the form that people or things show.

- Qualified: whether obtaining qualification certificate, medical license and logging qualification.

- Intention: the motivation of crime.
- Method: the means of crime.

Object: the items involved in the crime.

- Property: public and private property.
- Instrument: tools for criminal purpose.
- Animal: endangered animals.
- Plant: trees, such as pine tree, coconut tree.
- Drug: drugs, such as heroin, marijuana.
- Drug Plant: drug plants, such as opium poppy.
- Gambling Device: gambling tools.
- License: certificates and licenses.

Attribute: the abstract characterization of items, such as value, length.

- Quantity Attribute: measure words.
- Injury Attribute: the degree of injury.

C Details of Event Constraints

C.1 Absolute Constraint

During extraction trigger and role types, one subordinate trigger type must be extracted. And Criminal role type should appear at least once.

C.2 Consistency Constraint

A subordinate trigger type has its related subordinate role types. We show the corresponding in Table 11.

D Statistics of LJP-E

Table 12 shows the statistics of our proposed dataset LJP-E. There are 957 cases for training, 136 cases for validation and 274 cases for testing. The average number of arguments in a case is 4.13.

E Inter-Annotator Agreement

First, we measure inter-annotator agreement on trigger and argument annotations. Since these annotations involve annotation of text spans, we follow Araki et al. (2018), treating one person's annotations as gold and the other person's annotations as predicted and calculating the F1 score under two settings: strict matching and partial matching. The former measures whether two annotations have the exact same span. The latter measures whether there is an overlap between annotations. Note that partial matching has the restriction that each annotation can only be matched to one annotation by the other annotator. We then use the resulting F1 score as the inter-annotator agreement. The agreement scores are 0.8013 (strict) and 0.8425 (partial) for trigger annotation and 0.6728 (strict) and 0.7819 (partial) for argument annotation.

Next, we measure agreement on trigger type and argument role annotations. We follow the way we measured agreement on trigger and argument annotations, computing the F1 score using both strict matching and partial matching, where two annotations strictly match if both their spans and their types/roles exactly match, and two annotations partially match if they spans overlap and their types/roles are identical. The agreement scores are 0.7838 (strict) and 0.8255 (partial) for trigger type annotation and 0.6405 (strict) and 0.7332 (partial) for argument role annotation.

F Qualitative Analysis

Term of penalty error analysis. Compared to law article and charge prediction, the F1 score of term of penalty prediction is significantly low. Hence, we give an error analysis to show more insights of term penalty prediction subtask. Specifically, we calculate the error rate of the 11 penalty categories of *EPM* on the test portion of CAIL-small. As shown in Table 13, the category *5-7 years* has the highest error rate. In average, the model performs worse on severe penalty categories than mild penalty categories (take 3 years as the boundary). It may cause from the data imbalance problem, as there are fewer cases of severe penalty categories than that of mild penalty categories. Furthermore, we find that term of penalty prediction is significantly impacted by fine-grained information. For example, in two cases of intentional injury, the victim in one case is seriously injured, whereas the other case's victim is slightly injured. Though the other parts of the fact statements are similar, the final sentence is completely different (5 years for severe injury and 1 year for slight injury). Law article and charge prediction involve judging whether the fact statement matches with the event pattern in the relevant law article, whereas term of penalty prediction involves detecting event patterns and analyzing fine-grained information simultaneously, which make it more difficult. In the future, we will focus on improving term of penalty prediction.

Law article error analysis. We also give an error analysis of law article prediction. We select several cases for which *EPM* predicts wrong law articles. We find that *EPM* shows weakness in handling multiple events. If there are multiple events in the fact statement, *EPM* prefers to ex-

Subordinate Trigger Type	Subordinate Role Types
Robbery	Criminal, Victim, Property, Quantity
Theft	Criminal, Victim, Property, Quantity
Fraud	Criminal, Victim, Intention, Property, Quantity
Racketeering	Criminal, Victim, Intention, Property, Quantity
Intentional Injury	Criminal, Victim, Intention, Instrument, Injury
Rape	Criminal, Victim, Method
Kidnapping	Criminal, Victim, Method
Obstructing Official Duties	Criminal, Victim, Method, Officer, Injury
Forgery	Criminal, Intention, License, Quantity
Gambling	Criminal, Intention, Gambling Device, Quantity
Illegal Doctoring	Criminal, Qualified, License, Injury
Endangering Rare Wildlife	Criminal, Qualified, License, Animal, Quantity
Illegal Logging	Criminal, License, Qualified, Instrument, Plant, Quantity
Drug Possession	Criminal, Drug, Quantity
Illegal Planting Drug	Criminal, Qualified, License, Drug Plant, Quantity

Table 11: Trigger-Role.

Dataset	CAIL
#Training Set Cases	957
#Validation Set Cases	136
#Testing Set Cases	274
#Law Articles	15
#Charges	15
#Term of Penalty	11
#Average Arguments	4.13
#Superordinate Trigger	6
#Subordinate Trigger	15
#Superordinate Role	4
#Subordinate Role	16

Table 12: The statistics of LJP-E.

Penalty	Error Rate	# of Class
Death/life imprisonment	56.38%	0.39%
>10 years	38.44%	1.15%
7-10 years	70.50%	0.87%
5-7years	**95.16%**	1.19%
3-5 years	52.12%	3.77%
2-3 years	50.50%	6.95%
1-2 years	62.01%	11.46%
9-12 months	49.29%	16.11%
6-9 months	84.55%	12.73%
0-6 months	14.09%	41.83%
0 month	81.86%	3.50%

Table 13: The error rate of penalty. *# of Class* is the percentage of each class in the test portion of CAIL-small. *Error Rate* is the error percentage of each class.

tract the event type with more training data. For example, kidnapping and racketeering cases may contain intentional injury facts. There are more training samples of intentional injury than that of kidnapping/racketeering. When predicting law articles for kidnapping/racketeering cases with intentional injury facts, *EPM* tends to extract the event of intentional injury and ignore the events of kidnapping/racketeering, which leads to wrong predictions. It gives us motivation to solve the problem of multiple events in the future work.

Fact Statement: 被告人宋某于2016年3月29日，在京沪高速服务区卫生间内产下一名男婴，并将该名男婴遗弃在卫生间内离开... (The criminal Song gave birth to a baby boy in the bathroom of the Beijing-Shanghai Expressway Service Area at about 9:30 on March 29, 2016, and abandoned the baby boy in the bathroom...) **TOPJUDGE**: Article 261; Crime of child trafficking; 1-2 years imprisonment **EPM**: Article 261; Crime of abandoning babies; 9-12 moths imprisonment **Ground-truth**: Article 261; Crime of abandoning babies; 9-12 moths imprisonment
Fact Statement: 2013年5月，李某为参加工程招标，联系他人为其伪造了名称为“浙江大有构件有限公司”营业执照... (In May 2013, in order to participate in the project bidding, Li contacted others to forged a business license named "Zhejiang Dayou Component Co., Ltd."...) **TOPJUDGE**: Article 280; Crime of forging national agency certificates; 6-9 moths imprisonment **EPM**: Article 280; Crime of forging business unit certificates; 0-6 moths imprisonment **Ground-truth**: Article 280; Crime of forging business unit certificates; 0-6 moths imprisonment

Table 14: Examples of inconsistent predictions.

Charge error analysis. Obviously, the main charge prediction errors of *EPM* come from the wrong predicted law articles, as the predicted charge is dependent on the predicted law article by the cross-task constraint. Another type of errors comes from the weakness in discriminating charges with subtle difference, such as illegal hunting and illegal fishing, illegal hunting and endangering rare wildlife.

Consistency analysis. Recall that we propose cross-task constraints to reduce inconsistent predictions. The SOTA model *TOPJUDGE* also exploits the dependency between different subtasks. *TOPJUDGE* uses prediction results as auxiliary features, which can not guarantee the predictions

of subtasks are consistent. Different from *TOPJUDGE*, our cross-task constraints can ensure that the outputs are consistent by forcing the model to predict allowable charges or terms of penalty based on the predicted articles during inference. To analyze the impacts of cross-task constraints, we select cases from the test portion of CAIL for which *TOPJUDGE* predicts the correct law articles, but wrong charges or terms of penalties, whereas *EPM -w/o event* predicts the correct law articles, charges and terms of penalty. We find that *TOPJUDGE* without cross-task constraints may predict the wrong charges that are not allowable regrading the predicted law article. For example, as shown in Table 14, *TOPJUDGE* predicts child trafficking for an abandoning babies case, whereas *EPM -w/o event* exploits the cross-task constraint to select results among allowable candidates during inference and predicts correctly.

As for term of penalty prediction, *TOPJUDGE* has high possibility to predict severe penalty. In cases where the same law article applies, the proportion of large penalty values is higher than that of small penalty values. Thus, models may inherit this data bias and predict more severe penalty. As shown in Table 14, *TOPJUDGE* predicts 1-2 years imprisonment for a abandoning babies case, but its ground-truth penalty is 9-12 months imprisonment, whereas *EPM -w/o event* exploits cross-task constraints to predict allowable penalty regarding the predicted law article and alleviate the data bias problem.

Event impacts To analyze the impacts of predicted event information, we select 200 cases with case types belonging to LJP-E annotation inventory from the tesing portion of CAIL. For these 200 cases, *TOPJUDGE* predicts wrong law articles whereas *EPM -w/o all DEP* predicts correct law articles. We find that there are two main types of errors predicted by *TOPJUDGE* without event extraction. The first is "specialisation over generalisation", which is completely incorrect. To be specific, *TOPJUDGE* may focus on some words that strongly trigger wrong law articles. As shown in Table 15, in a case of illegal logging, the facts describe that *two criminals negotiated to cut down tree by discussion*. *negotiated* and *by discussion* strongly imply it is a contract crime related case. Hence, *TOPJUDGE* wrongly predicts the *Crime of contract fraud*. In contrast, *EPM* can overcome this bias and predict results based on events instead of single words. The other type of errors is "multiple choice", which is partially incorrect. A case may contain several facts that violate different law articles, which is a problem of dealing with multiple crimes. *TOPJUDGE* prefers to predicting one possible article without referring to global information. As shown in Table 15, a illegal doctoring case wrongly predicted by *TOPJUDGE* as a intentional injury case because of the fact that the victim was assessed as *a serious injury grade II*. In contrast, *EPM* exploits event information and predicts the correct article in global (note that a illegal doctoring event contains injury degree arguments that defined as role type "Injury"). Moreover, which wrong law articles does *TOPJUDGE* prefer to predict in the second error type? We find that *TOPJUDGE* prefers to predict the wrong law article having similar texts with the ground-truth. For example, the illegal doctoring law article and the intentional injury law article all have the texts about injury degree.

Fact Statement: 2014年4月，刘某和王某（另案处理）经过商量决定砍伐大新公司所种植的桃树，并协商好由王某负责销售。被告人刘某砍伐桃树达一百多吨... (In April 2014, Liu and Wang (handled in a separate case) decided to cut down the peach trees planted by Daxin Company after discussion, and negotiated that Wang would be responsible for the sale. Liu cut down 100 tons of peach trees...) **TOPJUDGE**: Article 224 [Crime of contract fraud] **EPM**: Article 345 [Crime of illegal logging] **Extracted event**: 刘某(Liu) [Criminal], 王某(Wang) [Criminal], 砍伐(Cut down) [Trigger], 一百吨(100 tons) [Quantity], 桃(peach) [Plant] **Ground-truth**: Article 345 [Crime of illegal logging]
Fact Statement: 2015年8月19日，被告人邵某在未取得相关行医资格的情况下，给被害人黄某以注射玻尿酸的方式做整形术，导致被害人黄某某右眼失明。经鉴定，被害人黄某被评定为重伤二级。(On August 19, 2015, the criminal Shao gave the victim Huang a plastic surgery by injection of hyaluronic acid without obtaining the relevant medical qualifications. This resulted in the victim Huang's right eye blindness. After identification, Huang was assessed as a serious injury grade II) **TOPJUDGE**: Article 235 [Crime of intentional injury] **EPM**: Article 336 [Crime of illegal doctoring] **Extracted event**: 邵某(Shao) [Criminal], 黄某(Huang) [Victim], 整形(plastic surgery) [Trigger], 行医资格(medical qualifications) [License], 重伤二级(serious injury grade II) [Injury] **Ground-truth**: Article 336 [Crime of illegal doctoring]

Table 15: Examples of event impacts.

G Event Extraction Performance

We show the event extraction performance on LJP-E. The metrics are macro-precision (P), macro-recall (R) and macro-F1 (F1). Here, both text span

	Role			Trigger		
	F1%	**P%**	**R%**	**F1%**	**P%**	**R%**
EPM	52.59	49.88	55.69	50.37	47.52	53.75
w/o CSTR1	50.07	50.09	50.38	48.41	49.21	47.90
w/o CSTR2	48.97	49.18	49.24	47.17	48.45	46.19
w/o DEP1	50.52	51.08	50.13	49.96	49.29	50.82
w/o DEP2	51.97	49.56	54.82	48.97	47.39	50.82
w/o hierarchy	48.97	51.07	47.04	47.62	45.45	50.00
w/ auxiliary	51.73	49.16	54.94	49.81	47.13	53.35

Table 16: Event extraction results on LJP-E.

and trigger type/argument role should be predicted correctly simultaneously. From Table 16, *EPM* achieves the best results on both trigger extraction and role extraction. Removing the two event constraints (i.e., *w/o CSTR1* and *w/o CSTR2*) degrades the event extraction results and negatively impacts LJP.

H Preprocessing CAIL

We follow existing works (Xu et al., 2020; Yang et al., 2019) to preprocess the CAIL dataset. Specifically, we filter out the case samples with multiple applicable law articles and multiple charges. Also, we only keep law articles and charges that applicable to not less than 100 corresponding case samples. Note the frequency of each law article or charge is calculated on the training and validation portions. Besides, cases of second instance are removed. And we follow (Zhong et al., 2018) to divide the terms of penalty into non-overlapping intervals.

I Experimental Details

For training, we utilize the Adam optimizer with learning rate of 10^{-4} and the batch size is 32. The warmup step is 3000. Models are trained for a maximum of 20 epochs. For testing, we calculate Macro-F1 score for each subtask on the validation portion after each epoch and select the model with the maximum Macro-F1 score on the validation portion for testing each subtask. Note the CAIL-big subdataset has no validation portion and the validation portion of CAIL-small is employed. For LJP-E dataset, we run experiments 5 times and report the average results. For the hyperparameters, λ, in the loss function, the best setting is {0.5, 0.5, 0.4, 0.2, 0.1} for $\{\lambda_{t_a}, \lambda_{t_c}, \lambda_{t_p}, \lambda_r, \lambda_p\}$ for both CAIL-big and CAIL-small. The experiments have been performed on two Tesla V100 GPUs.

J Details of the Switch

During testing on CAIL, we train a BERT-based *Switch* model using Legal-BERT (Zhong et al., 2019) as backbone to make binary classification. We split the training portion of CAIL into two clusters. One only contains the 15 case types of our annotate dataset. The other contains other case types of CAIL. The input of *Switch* is a fact statement and the output is a binary value. The binary value determines whether to use the pretrained version of *EPM* or the fine-tuned version of *EPM*. Specifically, we takes the hidden vector of [CLS] token for making binary classification. we set batch size to 32 and epoch to 20. We take Adam as the optimize and the learning rate is 0.0001. After training, the Switch achieves 89.82% and 85.32% Accuracy on the testing portion of CAIL-big and CAIL-small respectively.

K Explanation of Event constraints in Detail

We list our event constrains and then explain them.

Absolute constraint. For a legal event, the trigger must be present. If the trigger is missing, we impose the following penalty:

$$\mathcal{P}_t = \sum_{\tilde{r}\in\mathcal{TG}} \sum_{i=1}^{l_f} s_r(x_i, \hat{y}^{\tilde{r}}_{(i)}) - \max_{i,\tilde{r}\in\mathcal{TG}}[s_r(x_i, \hat{y}^{\tilde{r}}_{(i)})] + |1 - \max_{i,\tilde{r}\in\mathcal{TG}}[s_r(x_i, \hat{y}^{\tilde{r}}_{(i)})]| \tag{17}$$

where $\mathcal{TG}$ is the trigger inventory. $s_r(x_i, \hat{y}^{\tilde{r}}_{(i)})$ is the predicted softmax probability of trigger $\tilde{r}$ for x_i.

In the equation, $\sum_{\tilde{r}\in\mathcal{TG}} \sum_{i=1}^{l_f} s_r(x_i, \hat{y}^{\tilde{r}}_{(i)})$ is the summation of probabilities of all trigger types in the sequence and $\max_{i,\tilde{r}\in\mathcal{TG}}[s_r(x_i, \hat{y}^{\tilde{r}}_{(i)})]$ is the maximum probability of all trigger types in the sequence. $|.|$ denotes the absolute operation. The operations before absolute operation ensure that all probabilities of trigger types except the max one should be close to 0. It forces the model to predict only one trigger. The absolute operation guarantees that the max probability should be close to 1 (note $s_r(x_i, \hat{y}^{\tilde{r}}_{(i)})$ is a value after softmax). It means at least one trigger should be predicted and its softmax probability should be close to 1. The all operations makes the model to predict exactly one trigger.

If a required role r is missing, we impose the following penalty:

$$\mathcal{P}_r = |1 - \max_i [s_r(x_i, \hat{y}^{\tilde{r}}_{(i)})]| \tag{18}$$

This equation ensures that the maximum probability of the required role in the sequence should be close to 1. It enforces the model to predict the required role at least once.

Event-Based consistency constraint. If a trigger type is detected, all and only its related roles should be detected. For example, if a Illegal Doctoring event is detected, no roles related to Illegal Logging should be predicted. If a trigger r is predicted, we impose the following penalty:

$$\begin{aligned} \mathcal{P}_e = & \sum_{\tilde{r} \in \mathcal{R}^+} |1 - \max_i [s_r(x_i, \hat{y}^{\tilde{r}}_{(i)})]| \\ & + \sum_{\tilde{r} \in \mathcal{R}^-} \sum_{i=1}^{l_f} s_r(x_i, \hat{y}^{\tilde{r}}_{(i)}) \end{aligned} \tag{19}$$

where $\mathcal{R}^+$ is the set of roles that should occur if r is predicted, $\mathcal{R}^-$ is the set of roles that cannot occur.

This equation ensures that the maximum probability of positive roles in the sequence should be close to 1. It means each positive role type should appear at least once. The equation also ensures that the probabilities of negative roles in the sequence should be close to 0. It means none of negative role types should be predicted. When applying this type of penalty, we first obtain the predicted trigger and then dynamically add the corresponding penalty into the loss function.