Legal Judgment Prediction via Topological Learning

Haoxi Zhong*, **Zhipeng Guo***, **Cunchao Tu**, **Chaojun Xiao**, **Zhiyuan Liu**†, **Maosong Sun**
Department of Computer Science and Technology
State Key Lab on Intelligent Technology and Systems
Institute for Artificial Intelligence, Tsinghua University, Beijing, China
{zhonghx18,gzp17,xiaocj16}@mails.tsinghua.edu.cn, tucunchao@gmail.com
{lzy,sms}@tsinghua.edu.cn

Abstract

Legal **J**udgment **P**rediction (**LJP**) aims to predict the judgment result based on the facts of a case and becomes a promising application of artificial intelligence techniques in the legal field. In real-world scenarios, legal judgment usually consists of multiple subtasks, such as the decisions of applicable law articles, charges, fines, and the term of penalty. Moreover, there exist topological dependencies among these subtasks. While most existing works only focus on a specific subtask of judgment prediction and ignore the dependencies among subtasks, we formalize the dependencies among subtasks as a Directed Acyclic Graph (DAG) and propose a topological multi-task learning framework, TOPJUDGE, which incorporates multiple subtasks and DAG dependencies into judgment prediction. We conduct experiments on several real-world large-scale datasets of criminal cases in the civil law system. Experimental results show that our model achieves consistent and significant improvements over baselines on all judgment prediction tasks. The source code can be obtained from https://github.com/thunlp/TopJudge.

1 Introduction

Legal **J**udgment **P**rediction (**LJP**) aims to predict the judgment results of legal cases according to the fact descriptions. It is a critical technique for the legal assistant system. On the one hand, LJP can provide low-cost but high-quality legal consulting services to the masses who are unfamiliar with legal terminology and the complex judgment procedures. On the other hand, it can serve as the handy reference for professionals (e.g., lawyers and judges) and improve their work efficiency.

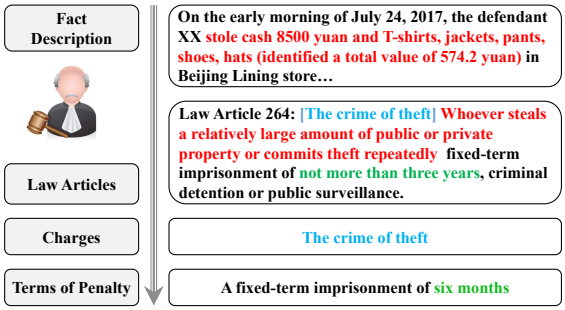


Figure 1: An illustration of the judicial logic of human judges in civil law system.

LJP has been studied for decades (Kort, 1957; Ulmer, 1963; Nagel, 1963; Keown, 1980; Segal, 1984; Lauderdale and Clark, 2012; Ye et al., 2018; Hu et al., 2018), and most existing works formalize LJP as a text classification task. For example, some works (Liu et al., 2004; Liu and Hsieh, 2006) propose to extract shallow textual features (e.g. characters, words, and phrases) for charge prediction. Katz et al. (2017) predict the US Supreme Court's decisions based on efficient features from case profiles. Luo et al. (2017) propose an attention-based neural model for charge prediction by incorporating the relevant law articles.

Despite these efforts in designing efficient features and employing advanced NLP techniques, LJP is still confronted with two major challenges:

Multiple Subtasks in Legal Judgment: Practically, legal judgment usually consists of detailed and complicated subclauses, such as charges, the term of penalty, and fines. Specifically, for those countries with the *civil law system* (e.g., China, France, and Germany), the prediction of relevant articles is also considered to be one of the fundamental subtasks, which will guide the prediction for other subtasks. In other words, all these subtasks compose the complete form of judgment pre-

∗ Indicates equal contribution. The order is determined by dice rolling.
† Corresponding author.

Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, pages 3540–3549
Brussels, Belgium, October 31 - November 4, 2018.

diction. Nevertheless, existing works on LJP usually focus on one specific subtask of judgments, which does not conform to the real scenarios. Although some methods (Luo et al., 2017) are developed to predict law articles and charges at the same time, their models are designed for a specific set of subtasks which are hard to scale to other subtasks.

Topological Dependencies between Subtasks: For human judges, there exists a strict order among the subtasks of legal judgment. As illustrated in Fig. 1, given the fact description of a specific case, a judge in the civil law system first decides which law articles are relevant to the scenario, and then determines the charges according to the instructions of relevant law articles. Based on these results, the judge further confirms the term of penalty and fines. How to simulate the judicial logic of human judges and model the topological dependencies among legal subtasks will deeply influence the creditability and interpretability of judgment prediction.

As stated above, conventional works cannot handle these two challenges due to both the limitation of specific tasks and neglecting topological dependencies. To address these issues, we propose to model the multiple subtasks in judgment prediction jointly under a novel multi-task learning framework.

We model the topological dependencies among these subtasks with a Directed Acyclic Graph (DAG), which means all subtasks are arranged in topological order. If the judgment of the j-th subtask t_j depends on the output of the i-th subtask t_i, then t_i appears earlier than t_j in such order. It is notable that such formulation provides an explicit explanation of dependency relations among subtasks.

Accordingly, we introduce topological learning for LJP and propose a unified framework, named as TOPJUDGE. Specifically, given the encoded representation of the fact description, TOPJUDGE predicts the outputs of all the subtasks following the topological order, and the output of a specific subtask will be affected by all the subtasks it depends on. In contrast with conventional multi-task learning, our model takes the explicit topological dependencies of LJP subtasks into consideration and is flexible to handle other LJP subtasks. Moreover, the topological order of legal dependencies renders our model interpretable and reliable.

To verify the effectiveness and flexibility of TOPJUDGE, we conduct a series of experiments on several real-world large-scale datasets. Experimental results show that our model achieves significant and consistent improvements over state-of-the-art models on all tasks and datasets. To summarize, we make several noteworthy contributions as follows:

(1) We are the first to explore and formalize the multiple subtasks of legal judgment under a joint learning framework. Moreover, we formulate the dependencies among the subtasks of LJP as a form of DAG and introduce this prior knowledge to enhance judgment prediction.

(2) We propose a novel judgment prediction framework, TOPJUDGE, to unify multiple subtasks and make judgment predictions through topological learning. This model can handle any form of DAG dependent subtasks, which has been verified in the experiments.

(3) We carry out experiments on several large-scale real-world datasets, and our model significantly and consistently outperforms all the baselines on all subtasks.

2 Related Work

2.1 Judgment Prediction

Employing automatic analysis techniques for legal judgment has drawn attention from researchers in the legal field for decades. Early works usually focus on analyzing existing legal cases in specific scenarios with mathematical and statistical algorithms (Kort, 1957; Ulmer, 1963; Nagel, 1963; Keown, 1980; Segal, 1984; Lauderdale and Clark, 2012).

With the development of machine learning and text mining techniques, more researchersformalize this task under text classification frameworks. Most of these studies attempt to extract efficient features from text content (Liu and Hsieh, 2006; Lin et al., 2012; Aletras et al., 2016; Sulea et al., 2017) or case annotations (e.g., dates, terms, locations, and types) (Katz et al., 2017). However, these conventional methods can only utilize shallow textual features and manually designed factors, both require massive human efforts and usually suffer from the generalization issue when applied to other scenarios.

Inspired by the success of neural networks on NLP tasks (Kim, 2014; Baharudin et al., 2010; Tang et al., 2015), researchers began to handle LJP by incorporating neural models with legal knowl-

edge. For example, Luo et al. (2017) present an attention-based neural network that jointly models charge prediction and relevant article extraction. Hu et al. (2018) incorporate 10 discriminative legal attributes to predict few-shot and confusing charges. Nevertheless, these models are designed for specific subtasks and thus non-trivial to be extended to more subtasks of LJP with complex dependencies. Besides, Ye et al. (2018) utilize a Seq2Seq model to generate court views with fact descriptions and predicted charges in Chinese civil law.

2.2 Multi-task Learning

Multi-task learning (MTL) aims to exploit the commonalities and differences across relevant tasks by solving them at the same time. It can transfer useful information among various tasks and has been applied to a wide range of areas, including NLP (Collobert and Weston, 2008), speech recognition (Deng et al., 2013), and computer vision (Girshick, 2015; Mao et al., 2017).

There have been numerous successful usages of MTL in NLP tasks. Most works follow the *hard parameter sharing* setting by sharing representations or some encoding layers among relevant tasks. For example, Collobert and Weston (2008) use shared word embeddings in solving part-of-speech tagging and semantic role labeling tasks. Liu et al. (2015) share the encoding layers of input queries to address query classification and information retrieval. Dong et al. (2015) and Luong et al. (2016) propose to share encoders or decoders to improve one (many) to many neural machine translation. Firat et al. (2016) propose to share attention mechanism in multi-way, multilingual machine translation. Besides *hard parameter sharing*, *soft parameter sharing* is another common approach in MTL. It assumes that each task owns its specific parameters and the distance between parameters in different tasks should be close to each other. For example, Duong et al. (2015) employ L2 distance for regularization, while Yang and Hospedales (2017) use the trace norm. Liu et al. (2016) introduce gates among task-specific RNN layers to control the information flow. Ruder et al. (2017) introduce a model which can decide the amount of sharing between different NLP tasks. There are also some works focusing on increasing tasks (Hashimoto et al., 2017) or handing unlabeled data (Augenstein et al., 2018).

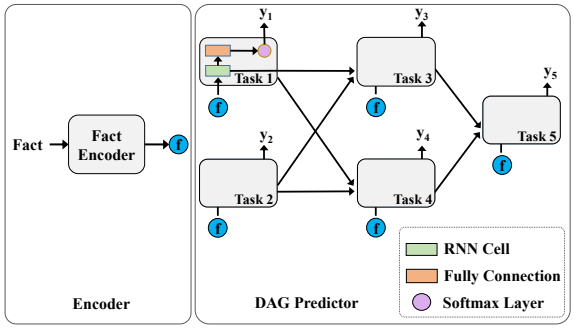


Figure 2: The framework of TOPJUDGE.

In this work, we introduce a topological learning framework TOPJUDGE to handle multiple subtasks in LJP. Different to conventional MTL models which focus on how to share parameters among relevant tasks, TOPJUDGE models the explicit dependencies among these subtasks with scalable DAG forms.

3 Method

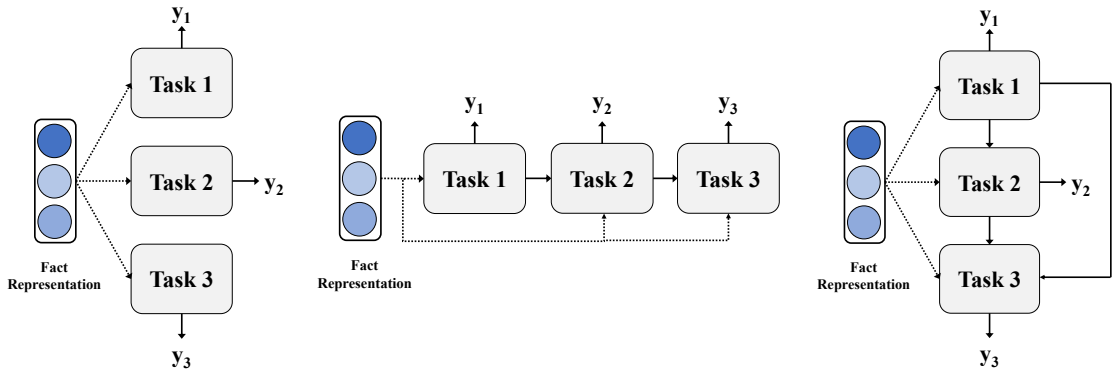
In the following parts, we will first give the essential definitions of LJP task. We then introduce the DAG dependencies of the subtasks in LJP. And finally, we describe the neural encoder for fact representation and the judgment predictor for the subtasks with DAG dependencies. The overall framework of TOPJUDGE has been shown in Fig 2.

3.1 Problem Formulation

We will focus on the LJP tasks in civil law. Suppose the fact description of a case is a word sequence $\mathbf{x} = \{x_1, x_2, \ldots, x_n\}$, where n is the length of $\mathbf{x}$ and each word x_i comes from a fixed vocabulary W. Based on the fact description $\mathbf{x}$, the task of LJP T aims to predict judgment results of applicable law articles, charges, term of penalty, fines and so on. Formally, we assume T contains $|T|$ subtasks, i.e., $T = \{t_1, t_2, \ldots, t_{|T|}\}$, each of which is a classification task. For the i-th subtask $t_i \in T$, we aim to predict the corresponding result $\mathbf{y}_i \subseteq Y_i$, where Y_i is a subtask-specific label set. Take the subtask of charge prediction for example, the corresponding label set should contain *Theft*, *Traffic Violation*, *Intentional Homicide* and so on.

3.2 DAG Dependencies of Subtasks

We assume that the dependencies among multiple subtasks of LJP form a DAG. As a result, the task list T should satisfy topological constraints. Formally, we use the notation $t_i \lhd t_j$ to denote that



(a) Typical MTL form of DAG (b) Sequential form of DAG (c) General form of DAG

Figure 3: Three typical forms of DAG dependencies.

the j-th subtask depends on the i-th subtask, and $D_j = \{t_i \mid t_i \lhd t_j\}$ to denote the dependency set. The task list T can be ordered to satisfy the following constraint

$$i < j, \quad \forall (i,j) \in \{(i,j) \mid t_i \in D_j\}. \tag{1}$$

We demonstrate the flexibility of our formulation by describing two special cases: (1) As shown in Fig. 3 (a), if no dependencies exist, i.e., $D_j = \emptyset$, it corresponds to the typical MTL setting where we simultaneously make predictions for all subtasks. (2) As shown in Fig. 3 (b), if each task only depends on its previous task, i.e., $D_j = \{t_{j-1}\}$, it forms a sequential learning process.

3.3 Neural Encoder for Fact Descriptions

We employ a fact encoder to generate the fact description's vector representation as the input of TOPJUDGE. In the following part, we briefly introduce an encoder based on Convolutional Neural Networks (CNN) (Kim, 2014).

Taking a word sequence $\mathbf{x}$ as input, the CNN encoder computes the text representation through three layers, i.e., lookup layer, convolution layer and pooling layer.

Lookup We first convert each word x_i in $\mathbf{x}$ into its word embedding $\mathbf{x}_i \in \mathbb{R}^k$, where k is the dimension of word embeddings. The word embedding sequence is then represented as

$$\hat{\mathbf{x}} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}. \tag{2}$$

Convolution A convolution operation involves a convolution matrix $\mathbf{W} \in \mathbb{R}^{m \times (h \times k)}$, which is applied to a sliding window of length h with number of filters m to produce a feature map by

$$\mathbf{c}_i = \mathbf{W} \cdot \mathbf{x}_{i:i+h-1} + \mathbf{b}, \tag{3}$$

where $\mathbf{x}_{i:i+h-1}$ is the concatenation of word embeddings within the i-th window and $\mathbf{b} \in \mathbb{R}^m$ is the bias vector. By applying convolution over each window, we obtain $\mathbf{c} = \{\mathbf{c}_1, \ldots, \mathbf{c}_{n-h+1}\}$.

Pooling We apply per-dimension max-pooling over $\mathbf{c}$ and obtain the final fact representation $\mathbf{d} = [d_1, \ldots, d_m]$ by

$$d_t = \max(\mathbf{c}_{1,t}, \ldots, \mathbf{c}_{n-h+1,t}), \forall t \in [1, m]. \tag{4}$$

3.4 Judgment Predictor over DAG

Based on the DAG assumption, we obtain an ordered task list $T^* = [t_1, t_2, \ldots, t_{|T|}]$. For each task $t_j \in T$, we aim to predict its judgment result $\mathbf{y}_j$ based on the fact representation vector $\mathbf{d}$ and the judgment results of its dependent tasks.

For prediction, we employ a specific LSTM cell for each task and get the output of each task in the topological order. More specifically, for each task $t_j \in T$, we obtain its final judgment result through three steps, i.e., cell initialization, task-specific representation, and prediction.

Cell Initialization As stated above, the prediction result of t_j will be conditioned on the fact representation $\mathbf{d}$ and the outputs of all dependent tasks $\mathbf{y}_k, \forall t_k \in D_j$. Hence, we have

$$\begin{bmatrix} \bar{\mathbf{h}}_j \\ \bar{\mathbf{c}}_j \end{bmatrix} = \sum_{t_i \in D_j} \left(\mathbf{W}_{i,j} \begin{bmatrix} \mathbf{h}_i \\ \mathbf{c}_i \end{bmatrix} \right) + \mathbf{b}_j \tag{5}$$

Here, $\mathbf{h}_i$ and $\mathbf{c}_i$ are the hidden state and memory cell of t_i. $\bar{\mathbf{h}}_j$ and $\bar{\mathbf{c}}_j$ are the initial hidden state and memory cell of t_j. $\mathbf{W}_{i,j}$ and $\mathbf{b}_j$ are transformation matrices and bias vectors specific to t_i and t_j.

Task-Specific Representation Taking the fact representation $\mathbf{d}$, the initial hidden state $\bar{\mathbf{h}}_j$, and the initial memory cell $\bar{\mathbf{c}}_j$ as inputs, we process

them with an LSTM cell (Hochreiter and Schmidhuber, 1997).

We regard the final hidden state $\mathbf{h}_j$ as the task-specific representation of task t_j. The last cell state $\mathbf{c}_j$ is used to compose the initial hidden state for the downstream tasks by Eq. 5

Prediction With the representation $\mathbf{h}_j$, we apply an affine transformation followed by softmax and obtain the final prediction as

$$\hat{\mathbf{y}}_j = \text{softmax}\left(\mathbf{W}_j^p \mathbf{h}_j + \mathbf{b}_j^p\right). \quad (6)$$

Here, $\mathbf{W}_j^p$ and $\mathbf{b}_j^p$ are parameters specific to task t_j.

With the prediction result $\hat{\mathbf{y}}_j$, we minimize the cross-entropy between $\hat{\mathbf{y}}_j$ and $\mathbf{y}_j$ as follows:

$$\mathcal{L}_j(\hat{\mathbf{y}}_j, \mathbf{y}_j) = -\sum_{k=1}^{|Y_j|} \mathbf{y}_{j,k} \log(\hat{\mathbf{y}}_{j,k}). \quad (7)$$

3.5 Training

We use cross-entropy loss for each subtask and sum up losses to train TOPJUDGE:

$$\mathcal{L} = \sum_{j=1}^{|T|} \lambda_j \mathcal{L}_j(\hat{\mathbf{y}}_j, \mathbf{y}_j), \quad (8)$$

where λ_j is the weight factor for each subtask t_j. The DAG dependencies of subtasks ensure that our model is differentiable and can be trained in an end-to-end fashion. In practice, we set all weights λ_j to 1, and employ Adam (Kingma and Ba, 2015) for optimization. We also apply dropout (Srivastava et al., 2014) on the fact representation to prevent overfitting.

4 Experiments

To evaluate the proposed TOPJUDGE framework, we conduct a series of experiments on LJP over three large-scale datasets of criminal cases in China. We select three representative judgment prediction subtasks for comparison, including law articles, charges, and the terms of penalty.

4.1 Dataset Construction

As there are no publicly available LJP datasets in previous works, we collect and construct three different LJP datasets, including **CJO**, **PKU**, and **CAIL**. CJO consists of criminal cases published by the Chinese government from China Judgement Online[1]. PKU contains criminal cases published by Peking University Law Online[2]. CAIL (Chinese AI and Law Challenge) is another criminal case dataset for competition released by the Supreme People's Court of China[3]. The details of CAIL can be found in Xiao et al. (2018).

Datasets	CJO	PKU	CAIL
Cases	1,007,744	175,744	113,536
Law Articles	98	68	105
Charges	99	64	122
Term of Penalty	11	11	11

Table 1: The statistics of different datasets.

For all datasets we mentioned above, as the documents are well-structured and human-annotated, we can easily extract *fact descriptions*, *applicable law articles*, *charges* and *the terms of penalty* from each document using regular expressions. We have manually checked a randomly sampled set of cases, and extraction errors are negligible.

In real-world scenarios, there are some cases with multiple defendants and multiple charges, which will increase the complexity of judgment prediction. As our model aims to explore the effectiveness of considering topological dependencies between various subtasks, we filter out these cases and leave them as our future work.

Meanwhile, there are also some infrequent charges and law articles, such as *money laundering*, *smuggling of nuclear materials* and *tax dodge*. We filter out these infrequent charges and law articles and only keep those with frequencies greater than 100. For the term of penalty, we divide the terms into non-overlapping intervals. We list detailed statistics of these datasets in Table 1.

4.2 Baselines

For comparison, we employ the following text classification models and judgment prediction methods as baselines:

TFIDF+SVM: We employ term-frequency inverse document frequency (TFIDF) (Salton and Buckley, 1988) to extract word features and utilize SVM (Suykens and Vandewalle, 1999) for text classification.

CNN: We employ CNN with multiple filter widths (Kim, 2014) for fact encoding and classification.

Hierarchical LSTM (HLSTM): Tang et al. (2015) employs hierarchical neural networks to learn document representations in sentiment clas-

1 http://wenshu.court.gov.cn/
2 http://www.pkulaw.com/
3 http://cail.cipsc.org.cn/index.html

	Tasks	Law Articles				Charges				The Term of Penalty			
	Metrics	Acc.	MP	MR	F_1	Acc.	MP	MR	F_1	Acc.	MP	MR	F_1
Single	TFIDF+SVM	82.4	45.5	26.7	30.2	82.2	47.4	27.9	31.3	48.5	36.0	16.7	16.5
	CNN	92.5	46.9	38.4	40.0	92.3	41.2	32.3	33.7	57.4	35.6	22.2	22.7
	HLSTM	91.4	38.6	37.3	36.9	91.8	37.8	36.0	35.2	56.1	22.5	25.0	23.3
Multi	Fact-Law Att.	93.5	50.9	45.6	45.9	93.4	47.2	41.4	41.5	56.3	31.3	26.4	26.7
	PM	93.7	51.9	44.1	44.9	93.6	45.5	39.1	39.3	58.2	38.2	24.9	26.8
	CNN-MTL	94.3	53.0	46.0	46.9	94.1	48.5	41.7	42.5	58.7	39.9	28.8	29.4
	HLSTM-MTL	92.4	45.5	41.4	41.0	92.3	41.9	36.6	35.9	54.9	30.6	26.6	26.4
Ours	**TOPJUDGE**	**94.4**	**53.9**	**47.3**	**48.2**	**94.9**	**53.9**	**48.2**	**49.1**	**58.8**	**40.2**	**32.9**	**32.8**

Table 2: Judgment prediction results on CJO.

	Tasks	Law Articles				Charges				The Term of Penalty			
	Metrics	Acc.	MP	MR	F_1	Acc.	MP	MR	F_1	Acc.	MP	MR	F_1
Single	TFIDF+SVM	80.9	51.3	32.6	36.4	81.0	53.4	35.4	38.7	45.3	30.4	17.4	17.2
	CNN	93.1	64.3	52.6	54.3	93.3	61.9	49.3	51.1	57.6	24.1	23.1	23.3
	HLSTM	91.7	54.4	53.4	50.9	91.9	52.5	48.9	47.3	54.3	20.6	21.7	19.0
Multi	Fact-Law Att.	93.9	68.1	63.4	63.5	94.2	65.8	58.5	58.7	55.7	27.7	27.4	26.5
	PM	94.4	69.6	61.0	62.2	94.3	65.1	56.2	57.2	58.2	36.2	26.4	27.1
	CNN-MTL	95.0	73.8	64.9	66.0	95.0	70.7	60.6	61.7	**58.4**	36.0	28.7	28.9
	HLSTM-MTL	93.9	71.2	64.6	65.1	93.8	67.8	60.0	60.7	55.4	31.3	26.2	25.7
Ours	**TOPJUDGE**	**95.4**	**76.4**	**67.6**	**68.4**	**95.6**	**75.9**	**69.6**	**70.9**	57.8	**38.9**	**32.1**	**31.8**

Table 3: Judgment prediction results on PKU.

sification. Based on this work, we employ an LSTM for sentence representations and another one to obtain the representation of complete fact descriptions.

Fact-Law Attention Model: Luo et al. (2017) proposes a neural charge prediction model by capturing the interaction between fact descriptions and applicable laws with attention mechanism.

Pipeline Model (PM): To demonstrate the advantage of TOPJUDGE on modeling subtasks jointly, we also implement a pipelined method for comparison. Here, we train 3 separate CNN classifiers for law articles, charges, and term of penalty. For each subtask, the input is the concatenation of the fact representation and the embeddings for predicted labels of previous subtasks.

Besides, we compare our model with conventional MTL methods that do not consider the dependencies among subtasks as in Fig. 3 (a). These methods are denoted as **CNN-MTL** and **HLSTM-MTL**, where we implement the fact encoder as in Fig. 2 using CNN or HLSTM respectively.

4.3 Experimental Settings

As the case documents are written in Chinese with no space between words, we employ THULAC (Sun et al., 2016) for word segmentation. Afterward, we adopt the Skip-Gram model (Mikolov et al., 2013) to pre-train word embeddings on these case documents, with embedding size set to 200 and frequency threshold set to 25.

For all models, we set the fact representation and task-specific representation size to 256. Meanwhile, we set the maximum sentence length to 128 words and maximum document length to 32 sentences.

For training, the learning rate of Adam optimizer is 10^{-3}, and the dropout probability is 0.5. We also set the batch size to 128 for all models. We train every model for 16 epochs, and evaluate the final model on the testing set.

We employ *accuracy* (Acc.), *macro-precision* (MP), *macro-recall* (MR) and *macro-F_1* (F_1) as evaluation metrics. Here, the macro-precision/recall/F_1 are calculated by averaging the precision/recall/F_1 of each category.

4.4 Results and Analysis

We evaluate the performance on three LJP subtasks, including law articles (denoted as t_1), charges (denoted as t_2), and the terms of penalty (denoted as t_3). Experimental results are shown in Tables 2, 3, and 4. Note that, we implement TOPJUDGE with the dependency relationship in Fig. 3 (c), i.e.,

$$D_1 = \phi, D_2 = \{t_1\}, D_3 = \{t_1, t_2\}. \quad (9)$$

	Tasks	Law Articles				Charges				The Term of Penalty			
	Metrics	Acc.	MP	MR	F_1	Acc.	MP	MR	F_1	Acc.	MP	MR	F_1
Single	TFIDF+SVM	60.1	54.9	45.3	46.3	59.2	53.9	45.0	45.7	28.4	22.9	20.0	18.1
	CNN	81.4	74.4	64.1	65.7	80.7	77.3	65.5	67.2	28.8	34.7	27.8	28.6
	HLSTM	-	-	-	-	-	-	-	-	-	-	-	-
Multi	Fact-Law Att.	70.9	64.8	63.6	59.1	68.7	66.1	65.3	60.1	36.5	29.9	27.6	27.1
	PM	84.7	80.7	68.6	70.8	83.6	81.6	70.0	72.1	**40.0**	**37.4**	32.0	31.6
	CNN-MTL	84.5	80.0	68.1	70.3	83.4	81.6	69.1	71.6	39.5	37.2	32.3	31.3
	HLSTM-MTL	-	-	-	-	-	-	-	-	-	-	-	-
Ours	**TOPJUDGE**	**86.3**	**81.9**	**71.1**	**73.4**	**85.7**	**83.4**	**76.0**	**78.3**	38.3	36.1	**33.1**	**32.1**

Table 4: Judgment prediction results on CAIL. Note that, "-" means the model does not converge within 128 epochs.

This means that the prediction of charges depends on law articles, and the prediction of term of penalty depends on both law articles and charges. Such explicit dependencies conform to the judicial logic of human judges, which will be verified in later sections. These results show that:

(1) The proposed TOPJUDGE model outperforms other baselines significantly on most subtasks and datasets. It demonstrates the effectiveness and robustness of our proposed framework.

(2) Compared with conventional single-task models, e.g., CNN and HLSTM, MTL methods take advantage of the correlation among relevant subtasks and thus achieve promising improvements. It indicates the importance of modeling LJP subtasks jointly.

(3) Moreover, TOPJUDGE significantly outperforms typical MTL models, especially on the prediction of charges and the terms of penalty. It verifies the rationality and importance of modeling dependencies over LJP subtasks with DAG.

4.5 Ablation Analysis

Tasks	t_1		t_2		t_3	
Metrics	Acc.	F_1	Acc.	F_1	Acc.	F_1
TOPJUDGE	**95.4**	**68.4**	**95.6**	**70.9**	**57.8**	**31.8**
- $t_3 \lhd t_1$	95.2	67.7	95.4	70.3	57.4	31.2
- $t_2 \lhd t_1$	94.8	64.7	94.9	60.2	57.0	31.6
ϕ	94.7	64.4	94.9	60.1	57.8	27.6

Table 5: Ablation analysis on PKU.

To further illustrate the significance of legal dependencies and explore how the DAG dependencies influence the performance, we evaluate the performance of TOPJUDGE under various DAG architectures. Using Eq. 9 as the full dependencies, we remove the dependency of $t_3 \lhd t_1$ (law articles and term of penalty, corresponding to the sequential form in Fig. 3), $t_2 \lhd t_1$ (law articles and charges), and all dependencies respectively. Results are summarized in Table 5.

We observe that the performance of TOPJUDGE decreases on all tasks after removing either dependency. More specifically, when we dropped dependencies $t_3 \lhd t_1$ and $t_2 \lhd t_1$ respectively, significant decreases are observed for t_3 and t_2 correspondingly. This demonstrates that incorporating dependencies is beneficial for relevant subtasks, verifying its guiding role in the civil law system.

Meanwhile, we note that there are two main differences between TOPJUDGE and traditional multi-task models, namely the **Cell Initialization** and the **Task-Specific Representation**. We can see that if we eliminate **Cell Initialization** from TOPJUDGE, the dependencies will not be represented in the model and it will become similar to **CNN-MTL**. If we eliminate the **Task-Specific Representation** from TOPJUDGE, TOPJUDGE will become the same as the **Pipeline Model**. In a word, the main improvement of our models comes from the combination.

4.6 Case Study

We give some intuitive examples to demonstrate the significance of TOPJUDGE on LJP subtasks.

As shown in Table 6, case 1 is about negligently causing a fire. The fact description of this case states "The defendant pulled up weeds in the fields and piled them up in haphazard stacks. Afterward, he lighted them up and triggered the forest fires..." TOPJUDGE predicts all judgments correctly, while CNN-MTL fails to predict the charge and term of penalty. Moreover, CNN-MTL obtains conflicting judgments, i.e., "*crime of arson*" and "*1-2 years*", due to its neglecting of dependencies of these subtasks. According to the legal provisions of law article 115, the crime of arson should be sentenced to more than 10 years.

Methods	Law	Charge	Term (months)
CNN-MTL	115	Arson(×)	12-24(×)
TOPJUDGE	115	Negligently Causing a Fire	0-6
CNN-MTL	293	Intentional Damage to Property(×)	12-24(×)
TOPJUDGE	293	Affray	0-6

Table 6: Example cases and their prediction results.

Case 2 in Table 6 is another evidence of the insufficiency of conventional MTL on LJP. This case is about picking quarrels and provoking troubles. Both CNN-MTL and TOPJUDGE succeed to predict the relevant law articles (i.e., law article 293 of the crime of affray). However, CNN-MTL is confused between "*crime of affray*" and "*crime of intentional destruction or damage of properties*", two charges similar to each other. Conversely, TOPJUDGE can utilize the prediction result of law articles and consequently prevent this confusion.

To summarize, modeling the explicit dependencies among various subtasks can remarkably help the LJP model address the issue of predicting conflicting results.

4.7 Error Analysis

Prediction errors induced by our proposed model can be traced down into the following causes.

Data Imbalance. For the subtasks of law articles and charges, our model achieves more than 90% on *accuracy*, while only about 60% for *macro-F1*. This issue is much more severe on the subtask of the term of penalty, which our model yields a poor performance of only 30% *macro-F1*. The bad performance is mainly due to the imbalance of category labels, e.g., there are only a few training instances where the term is "*life imprisonment or death penalty*". Most judgment prediction approaches perform poorly (especially for *Recall*) on these labels as listed in Fig. 4. Instance weighting schemes can be introduced to address this issue in future works.

Incomplete Information. Following existing LJP works, we predict the final judgment according to the fact descriptions, which is incomplete as compared to the whole materials relevant to this case. In Chinese Law, there are certain circumstances under which the sentence can be shortened. For example, minors usually receive a lighter sentence, and those guilty of misdemeanors are allowed for a secured pending trial while paying a security deposit. However, such information is not included in the fact descriptions. The lack of such information also raises difficulties for judgment prediction, especially for the prediction of the term of penalty. In Fig. 4, we can see that the highest error rate comes from the cases with a short term of penalty. Our model fails to distinguish the cases with no penalty and those with 0-6 months term of imprisonment.

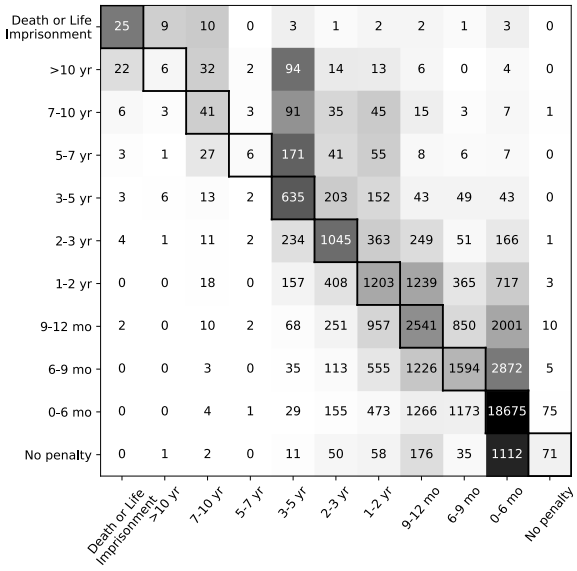


Figure 4: The confusion matrix in the subtask of predicting the term of penalty on the PKU dataset. The rows denote the ground truth while the columns denote the prediction results.

5 Conclusion

In this paper, we focus on the task of legal judgment prediction (LJP) and address multiple subtasks of judgment predication with a topological learning framework. To be specific, we formalize the explicit dependencies over these subtasks in a DAG form, and propose a novel MTL framework, TOPJUDGE, by integrating the DAG dependencies. Experimental results on three LJP subtasks and three different datasets show that our TOPJUDGE outperforms all single-task baselines and conventional MTL models consistently and significantly.

In the future, we will seek to explore the following directions: (1) We will explore more LJP subtasks and more scenarios of cases such as multiple

defendants and charges to investigate the effectiveness of TOPJUDGE. (2) We will explore how to incorporate into LJP the temporal factors, which are not considered in this work.

6 Acknowledgements

This work is supported by the National Natural Science Foundation of China (NSFC No. 61572273, 61772302) and the research fund of Powerlaw Inc. for AI+Law Technology. This work is also funded by China Association for Science and Technology (2016QNRC001). Tu is also supported by China Postdoctoral Innovative Talent Support Programme.

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.

Isabelle Augenstein, Sebastian Ruder, and Anders Søgaard. 2018. Multi-task learning of pairwise sequence classification tasks over disparate label spaces. In *Proceedings of NAACL*.

Baharum Baharudin, Lam Hong Lee, and Khairullah Khan. 2010. A review of machine learning algorithms for text-documents classification. *Journal of Advances in Information Technology*, 1(1):4–20.

Ronan Collobert and Jason Weston. 2008. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of ICML*, pages 160–167.

Li Deng, Geoffrey Hinton, and Brian Kingsbury. 2013. New types of deep neural network learning for speech recognition and related applications: An overview. In *Proceedings of ICASSP*, pages 8599–8603.

Daxiang Dong, Hua Wu, Wei He, Dianhai Yu, and Haifeng Wang. 2015. Multi-task learning for multiple language translation. In *Proceedings of ACL*, volume 1, pages 1723–1732.

Long Duong, Trevor Cohn, Steven Bird, and Paul Cook. 2015. Low resource dependency parsing: Cross-lingual parameter sharing in a neural network parser. In *Proceedings of ACL*, volume 2, pages 845–850.

Orhan Firat, Kyunghyun Cho, and Yoshua Bengio. 2016. Multi-way, multilingual neural machine translation with a shared attention mechanism. In *Proceedings of NAACL*, pages 866–875.

Ross Girshick. 2015. Fast r-cnn. In *Proceedings of ICCV*, pages 1440–1448.

Kazuma Hashimoto, Yoshimasa Tsuruoka, Richard Socher, et al. 2017. A joint many-task model: Growing a neural network for multiple nlp tasks. In *Proceedings of EMNLP*, pages 1923–1933.

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

Zikun Hu, Xiang Li, Cunchao Tu, Zhiyuan Liu, and Maosong Sun. 2018. Few-shot charge prediction with discriminative legal attributes. In *Proceedings of COLING*.

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

R Keown. 1980. Mathematical models for legal prediction. *Computer/LJ*, 2:829.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of EMNLP*.

Diederik Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *Proceedings of ICLR*.

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.

Benjamin E Lauderdale and Tom S Clark. 2012. The supreme court's many median justices. *American Political Science Review*, 106(4):847–866.

Wanchen Lin, Tsung Ting Kuo, and Tung Jia Chang. 2012. Exploiting machine learning models for chinese legal documents labeling, case classification, and sentencing prediction. In *Processdings of ROCLING*, page 140.

Chaolin Liu, Cheng Tsung Chang, and Jim How Ho. 2004. Case instance generation and refinement for case-based criminal summary judgments in chinese. *Journal of Informationence & Engineering*, 20(4):783–800.

Chaolin Liu and Chwen Dar Hsieh. 2006. Exploring phrase-based classification of judicial documents for criminal charges in chinese. In *Proceedings of ISMIS*, pages 681–690.

Pengfei Liu, Xipeng Qiu, and Xuanjing Huang. 2016. Recurrent neural network for text classification with multi-task learning. In *Proceedings of IJCAI*.

Xiaodong Liu, Jianfeng Gao, Xiaodong He, Li Deng, Kevin Duh, and Ye-Yi Wang. 2015. Representation learning using multi-task deep neural networks for semantic classification and information retrieval. In *Proceedings of NAACL*, pages 912–921.

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 EMNLP*.

Minh-Thang Luong, Quoc V Le, Ilya Sutskever, Oriol Vinyals, and Lukasz Kaiser. 2016. Multi-task sequence to sequence learning. In *Proceedings of ICLR*.

Jiayuan Mao, Tete Xiao, Yuning Jiang, and Zhimin Cao. 2017. What can help pedestrian detection? In *Proceedings of CVPR*, pages 3127–3136.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Proceedings of NIPS*, pages 3111–3119.

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

Sebastian Ruder, Joachim Bingel, Isabelle Augenstein, and Anders Søgaard. 2017. Learning what to share between loosely related tasks. *arXiv preprint arXiv:1705.08142*.

Gerard Salton and Christopher Buckley. 1988. Term-weighting approaches in automatic text retrieval. *Information processing & management*, 24(5):513–523.

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

Nitish Srivastava, Geoffrey E Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *JMLR*, 15(1):1929–1958.

Octavia Maria Sulea, Marcos Zampieri, Mihaela Vela, and Josef Van Genabith. 2017. Exploring the use of text classi cation in the legal domain. In *Proceedings of ASAIL workshop*.

Maosong Sun, Xinxiong Chen, Kaixu Zhang, Zhipeng Guo, and Zhiyuan Liu. 2016. Thulac: An efficient lexical analyzer for chinese.

Johan AK Suykens and Joos Vandewalle. 1999. Least squares support vector machine classifiers. *Neural processing letters*, 9(3):293–300.

Duyu Tang, Bing Qin, and Ting Liu. 2015. Document modeling with gated recurrent neural network for sentiment classification. In *Proceedings of EMNLP*, pages 1422–1432.

S Sidney Ulmer. 1963. Quantitative analysis of judicial processes: Some practical and theoretical applications. *Law and Contemporary Problems*, 28:164.

Chaojun Xiao, Haoxi Zhong, Zhipeng Guo, Cunchao Tu, Zhiyuan Liu, Maosong Sun, Yansong Feng, Xianpei Han, Zhen Hu, Heng Wang, et al. 2018. Cail2018: A large-scale legal dataset for judgment prediction. *arXiv preprint arXiv:1807.02478*.

Yongxin Yang and Timothy Hospedales. 2017. Deep multi-task representation learning: A tensor factorisation approach. In *Proceedings of ICLR*.

Hai Ye, Xin Jiang, Zhunchen Luo, and Wenhan Chao. 2018. Interpretable charge predictions for criminal cases: Learning to generate court views from fact descriptions. In *Proceedings of NAACL*.