Legal Knowledge and Information Systems
S. Villata et al. (Eds.)

doi:10.3233/FAIA200861

Integrating Domain Knowledge in AI-Assisted Criminal Sentencing of Drug Trafficking Cases

Tien-Hsuan WU [a], Ben KAO [a], Anne SY CHEUNG [b], Michael MK CHEUNG [b], Chen WANG [c], Yongxi CHEN [b], Guowen YUAN [a] and Reynold CHENG [a]
[a] *Department of Computer Science, The University of Hong Kong*
[b] *Faculty of Law, The University of Hong Kong*
[c] *College of Computer Science and Software, Shenzhen University*

Abstract. Judgment prediction is the task of predicting various outcomes of legal cases of which sentencing prediction is one of the most important yet difficult challenges. We study the applicability of machine learning (ML) techniques in predicting prison terms of drug trafficking cases. In particular, we study how legal domain knowledge can be integrated with ML models to construct highly accurate predictors. We illustrate how our criminal sentence predictors can be applied to address four important issues in legal knowledge management, which include (1) discovery of model drifts in legal rules, (2) identification of critical features in legal judgments, (3) fairness in machine predictions, and (4) explainability of machine predictions.

Keywords. judgment prediction, prison term prediction, domain knowledge, fairness, explainability

1. Introduction

With recent advances in machine learning (ML) and AI technology, one of the fastest growing areas in legal technology is the adoption of AI to assist lawyers and judiciaries in handling, processing, and discovering legal knowledge that is embedded in various legal documents such as judgments and ordinances. Works in this area have led to much interesting research, notably in *judgment prediction*, which is the task of predicting or determining various aspects of a legal case given a textual description of a litigation. Early works in judgment prediction (e.g., [1,2]) aim at predicting a certain outcome of a judgment by finding statistical correlations between a set of variables and possible outcomes from historical judgments. In recent years, researchers apply natural language processing (NLP) techniques and tackle the judgment prediction problem by formulating it as various text classification problems. Among them, the following four tasks have attracted much attention lately: **[Outcome prediction]**: to predict the outcome (e.g., settled, dismissed, etc.) of a case [3,4]; **[Article prediction]**: to identify relevant articles in law for a case [5,6,7]; **[Criminal charge prediction]**: to predict the charges of which a defendant should be convicted based on a description of the defendant's criminal activities [5,6,7,8]; and **[Prison term (sentence) prediction]** (or **PTP** for short): to predict the prison term to which a defendant should be sentenced [5,6,7,9]. In this paper we focus on PTP, which is the most challenging one of the four tasks due to the fact that criminal sentencing, as a form of judicial decision, may involve discretionary reasoning that is hard to specify as rules.

Most existing works in PTP build prediction models that take a textual description of a case as input and output a predicted sentence (prison term)[1]. These existing works generally suffer from the following inadequacies.

(Limited accuracy). Many of the works (e.g., [9,7,6]) formulate the PTP problem as a text classification problem in which (a textual description of) a case is classified into a group, each being associated with a prison-term range (e.g., “1-to-2 years”, “10 years or above”). The prediction is therefore imprecise. For works that predict a numerical value (e.g., prison term in months), the accuracy is generally not very high. For example, Zhong et al. [10] survey a number of predictors that participated in the Chinese AI and Law Challenge (CAIL2018). The predicted sentences made by the best predictor are on average about 38% off compared with the actual sentences.

(Little use of domain knowledge). Most existing works represent a case as unstructured text and apply NLP and neural networks to construct a sentence predictor. Although there are works that take legal domain knowledge into consideration, the application of which is very limited. For example, in Liu and Chen [9], only the average and the maximum imprisonment terms as stated in related law articles are used as domain knowledge in constructing the prediction models. We remark that legal domain knowledge can potentially help build more accurate models and thus should be effectively exploited.

(Non-explainability). The black-box model of neural networks employed by existing works does not provide sufficient information to explain sentencing decisions. Legal reasoning, however, is an important element in judgments. A sentence predictor should be able to explain the primary logic based on which a final sentence is made.

In this paper we study the PTP problem in the context of drug trafficking cases in Hong Kong. The reasons of our choice are twofold. First, Hong Kong has a common law system. Lower-level courts are bound to follow the rulings of appellate courts and the Court of Final Appeal, and to apply the law consistently. This makes prediction modeling based on historical judgments very applicable. Second, the sentencing of drug trafficking cases generally follows guidelines established in “tariff cases”. As we will discuss later, we take these guidelines as domain knowledge and show how they can be integrated into the construction of highly-accurate models. In our study, we take as input a judgment with sentencing information removed. The task of the predictor is to predict a final sentence of the case described in the judgment. Our major contributions are:

- We propose to tackle the PTP problem by integrating legal *domain knowledge* (DK) into ML modeling.
- We show how to apply ML techniques to (1) extract feature values from cases' textual descriptions given a feature set that is specified in the DK; and (2) construct various prediction models that consider the computational elements as specified in the DK.
- We show how our predictors can be applied to address a number of interesting legal/technical issues of PTP.

[1]We use the terms “sentence” and “prison term” interchangeably.

2. Related Work

Recent works formulate judgment prediction problems as text classification tasks, which take a case's textual description as input and predict an outcome using advanced NLP and ML techniques. For example, Vacek et al. [4] predict the outcomes of U.S. Federal Court judgments (such as dismissal by motion, settlement, etc.) and Hu et al. [8] predict the charges against a defendant. Both works use word embeddings to encode cases' descriptions which are then used to train various neural network models. Zhong et al. [6] and Yang et al. [7] use Long Short-Term Memory (LSTM) model to perform judgment prediction. In their works, they solve multiple judgment prediction tasks together to achieve synergetic effect. For example, the result of *article prediction* helps determine a defendant's *charge* as well as a range of the possible *prison term*.

Chen et al. [5] predict a prison term given a defendant's charges. In their work, a case's description and the charges are first encoded using embedding techniques. These embeddings are then fed into a Deep Gating Network (DGN) to determine a criminal sentence. One limitation of their approach is that numerical features, such as the value of stolen properties, are not sufficiently captured by their model. Moreover, the prediction model is trained using cases' textual descriptions. Hence, little legal domain knowledge is used. Liu et al. [9] also use embedding to encode cases. However, their predictor is informed with the ranges (min and max) of imprisonment terms stated in law articles.

3. Methodology

Given a textual description of a drug trafficking case, our task is to predict the prison term in number of months. In this section we describe our drug trafficking sentence predictors. In particular, we discuss the domain knowledge used and how it is integrated into our prediction modeling.

3.1. Data and Domain Knowledge (DK)

We start with a description of the legal data we use in training and evaluating our prediction models. We collected 3,172 English judgments on drug trafficking sentencing from Hong Kong courts. These judgments were handed down from 1998 to 2019. We consulted academic legal experts to identify two kinds of domain knowledge, namely, *substantive domain knowledge* (SDK) and *argumentative domain knowledge* (ADK). For SDK, our experts identified 6 categories of features that represent the most salient facts of a case. These categories are (1) *charge information* (e.g., name of charge and the related ordinances); (2) *drug information* (e.g., kind and weight of drugs involved); (3) *defendant background* (e.g., age, gender, nationality); (4) *mitigating factors* (e.g., guilty plea); (5) *aggravating factors* (e.g., persistent offender); (6) *sentence* (e.g., starting point sentence, final sentence). There are altogether 82 features. Our experts further identified 11 features (out of the 82) that are typically the determining factors of a sentence. These features are listed in Table 1[2]. We employed 11 law students to manually extract the values of all 82 features from each of the 3,172 judgments. As quality assurance, each

[2]We distinguish *guilty-plea* from the other mitigating factors because the sentence reduction for guilty-plea is quite standard.

Category	Feature Description
Drug Weights ($\mathcal{W}$)	weights of drugs (cocaine, heroin, ketamine, methamphetamine) involved
Plea ($\mathcal{P}$)	whether the defendant pleads guilty
Mitigating Factors ($\mathcal{M}$)	defendant shows **remorse**, drugs are mostly **self-consumed**, defendant assists in **controlled delivery**, defendant **gives testimony** in court, defendant has a **good character**
Aggravating Factors ($\mathcal{A}$)	defendant is a **refugee claimant**, defendant is **on bail**, defendant is a **persistent offender**, drugs are trafficked **internationally**

Table 1. Key factors in drug trafficking case sentencing

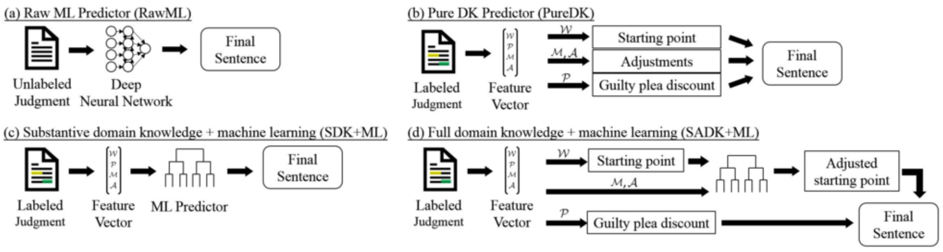

Figure 1. Predictors

judgment is processed by two workers to cross validate the extracted feature values. This "labeling task", which effectively transforms each piece of unstructured judgment text into structured data, took about 6 months to complete. Since our objective is to predict the prison term of a defendant given his/her offense, we remove cases that involve multiple defendants (so that the textual description given in a judgment focuses only on a given defendant). We also remove cases that involve very rare elements (such as rare drugs) because there are insufficient prior judgments to build reliable predictors for those cases. Our final set of judgments consists of 1,641 cases with an average prison term 91.4 months. The lengths of the judgments ranges from 115 words to 2,668 words, with the average being 475 words. ADK refers to a set of procedural rules to calculate the length of the prison term, as explained in detail in Section 3.2. Note that the manually labeled data serves as training data and test data for us to evaluate our prediction models. In Section 3.2.1 we will discuss how we train a machine to automatically extract features from legal documents, thus reducing the high cost of manual labeling.

3.2. Predictors

We consider four predictor models, which are illustrated in Figure 1. These predictors differ in whether and how ML and/or SDK/ADK are used.

Raw ML Predictor (RawML). The first predictor (Figure 1 (a)) does not use any domain knowledge. It takes as input a judgment (with sentencing information masked) and predicts the prison term using a deep neural network. Specifically, we follow the general architecture documented in Zhong et al. [10] to build the predictor — we encode each judgment as a sequence of word embeddings using word2vec [11]. The word embeddings are passed into Stacked Gated Recurrent Unit [12] to obtain a document-level vector, which is then passed to a fully connected layer with sigmoid activation for sentence prediction. We call this baseline approach *RawML* (for "Machine-learning-based with only raw data").

Pure DK Predictor (PureDK). The second predictor (Figure 1 (b)) follows closely how a human judge would determine a prison term. Both SDK and ADK are applied. For SDK, all the features listed in Table 1 are used in the computation. For ADK, the following 3-step procedure, as advised by legal experts, are carried out:
(1) *Starting point calculation:* For popular drugs, there are tariff cases in which sentencing guidelines are established to determine starting point penalty[3]. A typical guideline for a drug gives a list of *weight ranges* and for each range a *prison term range* (e.g., 10-50g of heroin → 5-8 years of imprisonment). Given a drug weight w that falls within a weight range $[w_1, w_2]$ with the corresponding starting point penalty range $[t_1, t_2]$, we assume a linear model in determining a starting point penalty *sp*, i.e., $sp = t_1 + (w - w_1)/(w_2 - w_1)$.

In many drug trafficking cases, however, more than one drug is involved. In this case, judges apply the *absurdity test*, the *conversion test*, and the *ratio test* to cross-check the appropriate sentence. We apply the *ratio test*[4] in our model to determine an aggregated starting point, which is computed as follows. Let w_1, ..., w_n be the weights of n types of drugs involved, and let $w_T = \sum_i w_i$ be the total weight. A judge would first compute the starting point penalty (sp_i) for type-i drug *as if all the drugs dealt were type-i* (i.e., sp_i is determined by the guideline of type-i drug with weight = w_T). The overall starting point *sp* is then given by a weighted sum of the sp_i's. Specifically, $sp = \sum_i (sp_i \times w_i / w_T)$.
(2) *Guilty plea discount*: If a defendant pleads guilty, a judge usually assesses a 1/3 discount to the penalty. We thus set a *guilty-plea-factor* (*gpf*) to $(1 - 1/3) = 2/3$ if the defendant pleads guilty; or 1 otherwise.
(3) *Adjustments*: Let $\mathcal{F} = \mathcal{M} \cup \mathcal{A}$ be the set of mitigating and aggravating factors (see Table 1). Note that we model each such factor $f_i \in \mathcal{F}$ as a binary feature, i.e., f_i is either present or absent in a case. For each f_i, a judge would assess a sentence adjustment (adj_i) if f_i is present. We estimate the adjustment adj_i by the average sentence reduction (if f_i is mitigating) or increment (if f_i is aggravating) observed in historical judgments due to factor f_i. An *adjustment factor* (*af*) is estimated by: $af = \prod_{f_i \text{is present}} (1 + adj_i)$.

The final predicted sentence (*fps*) is given by $fps = sp \times gpf \times af$. We call this predictor *PureDK* (for "Pure domain-knowledge-based without machine learning").

Substantive domain knowledge + machine learning (SDK+ML). The third predictor (Figure 1 (c)) is constructed by building regression trees with gradient boosting using the features listed in Table 1 as input and a prison term as output. We call this predictor *SDK+ML* as it utilizes substantive domain knowledge and ML techniques. Note that its construction is completely data-driven. In particular, it is not given any argumentative knowledge such as the *starting point guidelines*, *ratio test*, or *the guilty-plea discount*.

Full domain knowledge + machine learning (SADK+ML). The last predictor (Figure 1 (d)) uses all domain knowledge (SDK and ADK) plus machine learning techniques. We call it *SADK+ML*. It is essentially a hybrid of PureDK and SDK+ML. Similar to PureDK, SADK+ML uses the features specified in the SDK and it follows the procedure given in

[3] *R v Lau Tak-ming & Others* [1990] 2 HKLR 370; *HKSAR v Abdullah Anwar Abbas* [2009] 2 HKLRD 437; *Attorney General v Pedro Nel Rojas* [1994] 1 HKC 342; *HKSAR v Tam Yi-chun* CACC524/2011; *Secretary for Justice v Hii Siew Cheng* [2009] 1 HKLRD 1
[4] *HKSAR v Chan Yuk Leong* [2014] HKLRD (Yrbk) 325

the ADK in determining a starting point penalty, *sp*, and a guilty plea factor, *gpf* (see Steps (1) and (2) under PureDK). While PureDK assesses the adjustment of each mitigating and aggravating factor independently (based on the average adjustments observed in historical judgments), SADK+ML uses ML techniques to learn an overall adjustment model. Specifically, under SADK+ML, we build regression trees with gradient boosting that take starting point *sp*, mitigating factors ($\mathcal{M}$) and aggravating factors ($\mathcal{A}$) as input and predict an adjusted starting point penalty (*asp*). The reason for applying ML in determining adjustments is that occasionally judges do not articulate the adjustment of each factor, but determine an overall adjustment after considering all the factors.

3.2.1. Automatic Feature Extractor (AFE)

Our predictors, except for RawML, assume that features given in the SDK (Table 1) are extracted from judgments. These judgments are said to be "labeled" with the feature values identified. The feature values can be obtained manually, for example, by asking a user of the predictor to input drug types and weights, and to answer 10 yes/no questions for the 10 binary features. Alternatively, if the information of a case is given by a textual description (such as a plaintext judgment), we can apply machine comprehension to automatically extract feature values from text. We have implemented an automatic feature extractor (AFE). In this section we briefly describe the design of the AFE.

We use a combination of *regular expression* (RE) and *deep neural networks* (DNN) methods to extract features from judgments. Specifically, we use regular expressions to find drug types and weights, and guilty plea as the descriptions of these are relatively standard. As an example, from the text *"Defendant ... unlawfully trafficked in 3.56 kilograms of a solid* ***containing 2.29 kilograms of cocaine****"*, our RE extractor recognizes the pattern in boldface and correctly retrieves 2,290g (drug weight) and *cocaine* (drug type) from the text.

We use deep recurrent neural network [13] to classify text to determine the presence/absence of mitigating and aggravating factors. Specifically, a judgment is converted to a sequence of vectors (x_1, ..., x_n) using word2vec. The vectors are then sent to a network that consists of two stacked Bi-LSTM layers [14]:

$$y_i^{(1)} = [\overrightarrow{\text{LSTM}}(x_i);\overleftarrow{\text{LSTM}}(x_i)]; \quad y_i^{(2)} = [\overrightarrow{\text{LSTM}}(y_i^{(1)});\overleftarrow{\text{LSTM}}(y_i^{(1)})], \tag{1}$$

where $y_i^{(j)}$ denotes the output of the i-th vector in layer j. The output of the last unit in the Bi-LSTM layer is passed to a fully connected layer with sigmoid activation.

4. Performance

We conducted experiments to evaluate the four predictors. To recap, RawML takes a plaintext judgment (with sentencing masked) as input and returns a predicted prison term. It is oblivious to any legal guidelines or logic that a judge would usually follow. It serves as a baseline to illustrate how well a pure machine learning approach performs, which is also a typical approach taken by many existing works. PureDK mimics a human judge's decision by considering both substantive and argumentative knowledge. It calculates a prison term based on a feature vector (Table 1) and the three steps we previously described. PureDK thus provides the decision of a (pseudo) human judge as another base-

Predictor	RawML	PureDK	SDK+ML	SADK+ML	SDK+ML (AFE)	SADK+ML (AFE)
Accuracy (%)	73.03	91.52	91.23	92.12	88.04	88.90
$M_{0.3}$ (%)	33.19	4.33	5.91	5.55	9.69	7.86

Table 2. Predictors' accuracies

Factor	(a) # of Cases	(b) Sentence Adjustment	(c) Gini Importance
Show remorse	138	-0.62%	.0853
Self-consumption	222	-8.61%	.2301
Controlled delivery	36	-5.37%	.1605
Give testimony	18	-11.84%	.2243
Good character	24	-3.33%	.0125
Refugee claimant	23	+5.95%	.0392
On bail	29	+5.11%	.0019
Persistent offender	81	+5.91%	.0261
International	468	+5.19%	.2201

Table 3. Impact of factors on sentence prediction

line for comparison. SDK+ML and SADK+ML apply ML techniques and use DK to different extent — SDK+ML uses SDK to obtain a set of important features. The rest of the sentence prediction pipeline is completely done by ML. SADK+ML uses ADK to determine starting point and guilty-plea discount. However, it uses ML to learn a sentence adjustment model, which is not well promulgated in laws. For each of SDK+ML and SADK+ML, we consider two versions: one with feature values provided (by human labelers) and another one with feature values extracted by our AFE. The latter version represents the scenario of a fully automatic predictor that predicts a sentence by comprehending a plaintext case description.

All predictors are trained and evaluated with our corpus of 1,641 drug trafficking judgments (see Section 3.1) using 5-fold cross validation. Given a case, we measure a predictor's accuracy by $1 - (|\hat{y} - y|/y)$, where $\hat{y}$ is the predicted prison term and y is the prison term given in the corresponding judgment (i.e., ground truth). Also, we count the fraction of cases, denoted by M_a, in which a predictor's error $(|\hat{y} - y|/y)$ is at least a. For large a, say 30%, we consider the prediction a "big miss" (because of the substantial error). Table 2 shows the predictors' average accuracies and $M_{0.3}$. From the table, we make some observations.

• RawML's accuracy (73.03%) is much smaller than those of the other three predictors (which are in the 90's). Moreover, about 1/3 of RawML's predictions are big misses (predicted sentences are at least 30% off from ground truths). This shows that it is challenging for a pure machine learning approach to learn the models of hidden logic such as sentencing guidelines, ratio tests, and sentence adjustment.

• PureDK gives a very high accuracy (91.52%) and the lowest big-miss rate ($M_{0.3}$ = 4.33%). Recall that PureDK acts as a (pseudo) human judge by modeling the sentencing procedure based on the knowledge provided to us by human legal experts. The excellent performance of PureDK shows that the model is consistent with the decisions made by different human judges of the historical judgments. This infers that the human judges are very consistent in their judicial decisions on sentencing, closely following guidelines and common principles. We remark that our analysis of PureDK provides a data-driven scientific approach to studying the consistency issue in legal system, which would be otherwise difficult to perform considering the large number and big variety of cases.

• SDK+ML's accuracy (91.23%) is comparable to that of PureDK (91.52%) and it gives a bigger (5.91%) but still small big-miss rate. This shows that it is possible and effective to use ML techniques to perform PTP on drug trafficking cases. The fact that SDK+ML performs much better than RawML shows that substantive domain knowledge is critical to solving the PTP problem. Feature engineering is thus a necessary step in developing an effective solution.

• SADK+ML gives the best accuracy (92.12%), which is even slightly better than PureDK's (pseudo human judge). Recall that SADK+ML uses ML to learn a sentence-adjustment model. As we have previously mentioned, sentence adjustments are sometimes not perfectly articulated in judgments and that judges exercise discretion in determining an overall adjustment when there are multiple mitigating/aggravating factors. Our results show that ML techniques can be effectively applied to learn an aggregated adjustment model.
• Finally, the versions of SDK+ML and SADK+ML that use AFE to automatically extract feature values give very good accuracies and reasonably low $M_{0.3}$. Like RawML, the AFE versions of the predictors comprehend plaintext case descriptions. In particular, they significantly outperform RawML. This shows that our AFE is very effective.

5. Applications

We apply our sentence predictors to perform a number of interesting legal analytics studies. We discussed judgment consistency evaluation in the last section. In this section we briefly discuss four other applications of our predictors.

Model Drift. Our ML-based predictor learns a sentencing model from historical judgments. An implicit assumption is that those judgments follow the same hidden model. An interesting application of our predictor is to detect "model drift", which is a change of the hidden model, by detecting *outliers*. An outlier is a case whose sentencing deviates much from our model's prediction. Among the outlier cases, we find a specific case *C*, which is one of the most early cases in our corpus. It turns out that in case *C*, an unprecedentedly large amount of cocaine was dealt. Shortly after case *C*, a new guideline was laid down that has a binding effect on future similar cases. The model our predictor learns fits the new guideline well as most of the cases in our corpus are after case *C*. Since case *C* does not follow the new guideline, it is an outlier of our predictor. Hence, by outlier detection, our sentence predictor can detect model drifts. It also helps us identify appropriate sets of historical data for constructing models that are valid through different periods of time.

Factor Impact. With SDK, factors that would impact the final sentencing are identified by legal experts. An interesting question is what these factors' relative impacts are. We apply our predictor to provide a quantitative impact analysis.

We first study the impacts of the 9 mitigating/aggravating factors on final sentencing. Table 3, Column (b) shows the average sentence adjustment due to each factor. We see that the adjustments vary from the highest impact (*give testimony*, -11.84%) to the lowest (*show remorse*, -0.62%). The *testimony* factor has the highest impact because a defendant's testimony is often crucial in convicting accomplices and the mastermind of an offense. On the contrary, *remorse* has low impact because drug trafficking is a serious crime and the court seldom reduces sentence for such a minor factor. We remark that this impact analysis helps legal professional to statistically review the key factors in judicial decisions.

We further investigate how the factors impact the predictor's confidence. Table 3, Column (c) shows the normalized *Gini Importance (GI)* of each factor. The GI of a factor X quantifies how well we improve the predictor's confidence if X is known. The GI of factor X is measured by the reduction in the data impurity of each regression tree node that is split based on X, weighted by the probability of reaching that node in a

Feature	Group	# of cases	$\bar{e}$	σ_e	p-value
Gender	Male	995	0.0355	0.1891	0.4495
	Female	209	0.0448	0.1558	
Nationality	Local	90	0.0095	0.1095	0.6010
	Foreigner	219	0.0023	0.1090	
Age	Youth (16–20)	170	0.0435	0.2733	0.7476
	Aged 21+	1,090	0.0366	0.1462	

Table 4. Predictor's fairness evaluation

```
[Starting point]
   Heroin (482.82g): 204.9 months;
   Meth (14.38g): 87.5 months;
   Combined (ratio test applied): 209.4 months
[Adjustments]
   Give testimony in court: -11.84%;
   International element: +5.19%;
   Adjusted sentence: 194.2 months.
[Guilty plea]
   Yes (early stage): 1/3 sentence discount.
[Final sentence (predicted)]
   129.5 months.
```

Figure 2. Explanation of the prediction

prediction. From the table, we see that the most important factors in terms of GI are *self-consumption*, *give testimony*, and *international*. Note that even the *international* factor has a mild sentence impact (column (b)), it has a high GI because it is involved in many cases (column (a)). This knowledge allows us to selectively deploy more resources on extracting the high-GI factors, e.g., by multiple manual validations. Low-GI factors, on the other hand, can be more economically extracted by automatic extraction.

Fairness. *Algorithmic bias* refers to systematic errors made by an algorithm that produce unfair, favorable outcomes for one group of users/subjects over other groups. For example, a PTP predictor that tends to over-estimate the prison terms of male offenders and under-estimate those of female is biased. These biases should be avoided [15]. Fairness (absence of bias) is an important quality in an algorithmic decision system. Previous studies (e.g., [16]) show that even if demographic features are excluded in model construction, biases may still occur due to feature correlations. For example, from our data, the aggravating factor *international trafficking* (which is used in sentence prediction) and the demographic feature *nationality* are positively correlated.

We evaluate our predictor in terms of fairness by comparing the errors it makes over different groups of a demographic feature; if there is no statistically significant difference in the errors, the predictor treats the groups similarity and is thus fair. Specifically, for each drug trafficking case, we measure an error $e = (\hat{y} - y)/y$ where $\hat{y}$ is the predicted sentence and y is the actual sentence. Note that e can be positive (overestimate) or negative (underestimate). Table 4 shows the mean ($\bar{e}$) and standard deviation (σ_e) of e for different groups under three demographic features, namely, *gender*, *nationality*, and *age*[5]. These three features are studied because there are prior rulings that forbid biases with respect to them in sentencing[6]. We determine if there is a significant difference between the error means of two groups by Welch's t-Test [17]. A p-value > 0.05 indicates that the evidence of two groups having different means is weak; hence, the predictor is unbiased. From Table 4, we see that the average errors of different groups under each demographic feature are small and are sufficiently similar. Moreover, the p-values are all much larger than 0.05. The predictor is therefore fair.

Prediction Explainability. Besides improving prediction accuracy, the integration of domain knowledge (both SDK and ADK) allows us to design explainable models, which is very important in the legal domain as judges often explain their decisions in judgments. Note that RawML does not apply any domain knowledge and it is difficult to explain

[5]Cases for which a demographic feature is not documented in the judgments are not included in the data. *Race* is also an important factor for fairness evaluation, but it is not included because such information cannot be found from the judgments.

[6]*R v Okuya and Nevaboi* (1984) 6 Cr App R (S) 253, *HKSAR v Hong Chang Chi* [2002] 1 HKC 295, *R v Lau Tak-ming & Others* [1990] 2 HKLR 370

its logic from a legal perspective. In contrast, PureDK models the decision elements that judges generally follow and thus it provides *explainability by design*. To illustrate, Figure 2 is an example output of a sentence predictor we have developed. In this example, the predictor is given a case (*HKSAR v Kwan Yun-hang*) in which the defendant pleaded guilty for importing 482.82g of heroin and 14.38g of methamphetamine hydrochloride. The defendant helped the authority to convict other criminals by giving testimony in court. The sentence passed upon the defendant was 11 years (132 months). Note that our model's prediction of 129.5 months is very close to the actual sentencing.

6. Conclusion

In this paper we studied the prison term prediction (PTP) problem in the context of drug trafficking cases. We considered two kinds of domain knowledge, namely, substantive domain knowledge (SDK) and argumentative domain knowledge (ADK). We showed how the knowledge can be integrated with ML models to construct highly-accurate sentence predictors. Furthermore, we discussed a number of important applications of the predictors. Our study provides an example based on which similar techniques can be derived in other applications and legal domains.

References

[1] Nagel SS. Applying correlation analysis to case prediction. Texas Law Review. 1963;42:1006.

[2] Segal JA. Predicting Supreme Court cases probabilistically: The search and seizure cases, 1962-1981. American Political Science Review. 1984;78(4):891–900.

[3] Sulea OM, Zampieri M, Malmasi S, Vela M, Dinu LP, van Genabith J. Exploring the use of text classification in the legal domain. Proceedings of the Second Workshop on Automated Semantic Analysis of Information in Legal Text. 2017.

[4] Vacek T, Teo R, Song D, Nugent T, Cowling C, Schilder F. Litigation Analytics: Case outcomes extracted from US federal court dockets. In: Proceedings of the Natural Legal Language Processing Workshop 2019; 2019. p. 45–54.

[5] Chen H, Cai D, Dai W, Dai Z, Ding Y. Charge-Based Prison Term Prediction with Deep Gating Network. EMNLP. 2019.

[6] Zhong H, Guo Z, Tu C, Xiao C, Liu Z, Sun M. Legal judgment prediction via topological learning. In: EMNLP; 2018. p. 3540–3549.

[7] Yang W, Jia W, Zhou X, Luo Y. Legal Judgment Prediction via Multi-Perspective Bi-Feedback Network. Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence. 2019.

[8] Hu Z, Li X, Tu C, Liu Z, Sun M. Few-shot charge prediction with discriminative legal attributes. In: Proceedings of the 27th International Conference on Computational Linguistics; 2018. p. 487–498.

[9] Liu YH, Chen YL. A two-phase sentiment analysis approach for judgement prediction. Journal of Information Science. 2018;44(5):594–607.

[10] Zhong H, Xiao C, Guo Z, Tu C, Liu Z, Sun M, et al. Overview of CAIL2018: Legal Judgment Prediction Competition. arXiv preprint arXiv:181005851. 2018.

[11] Mikolov T, Sutskever I, Chen K, Corrado GS, Dean J. Distributed representations of words and phrases and their compositionality. In: Advances in neural information processing systems; 2013. p. 3111–3119.

[12] Chung J, et al. Gated feedback recurrent neural networks. In: ICML; 2015. p. 2067–2075.

[13] Pascanu R, Gulcehre C, Cho K, Bengio Y. How to construct deep recurrent neural networks. International Conference on Learning Representations. 2014.

[14] Graves A, Schmidhuber J. Framewise phoneme classification with bidirectional LSTM and other neural network architectures. Neural networks. 2005;18(5-6):602–610.

[15] Tonry M. Legal and ethical issues in the prediction of recidivism. Federal Sentencing Reporter. 2014;26(3):167–176.

[16] Tolan S, Miron M, Gómez E, Castillo C. Why Machine Learning May Lead to Unfairness: Evidence from Risk Assessment for Juvenile Justice in Catalonia. ICAIL. 2019.

[17] Welch BL. The generalization of 'Student's' problem when several different population variances are involved. Biometrika. 1947;34(1/2):28–35.