Manuscript accepted for the Ninth International Conference on Artificial Intelligence and Law, Edinburgh, Scotland, UK.

Classification and Clustering for Case-Based Criminal Summary Judgments

Chao-Lin Liu
Department of Computer Science
National Chengchi University
Taipei 11605, Taiwan
chaolin@nccu.edu.tw

Cheng-Tsung Chang
Department of Computer Science
National Chengchi University
Taipei 11605, Taiwan
g9005@cs.nccu.edu.tw

Jim-How Ho
Panchiao District Court
Taipei 236, Taiwan
g8905@cs.nccu.edu.tw

ABSTRACT

We investigate the effectiveness of machine-generated criteria for classification problems related to criminal summary judgments. Our system utilizes documents of closed lawsuits as training data for generating keyword-based and case-based classification criteria, and applies these machine-generated criteria for the classification tasks. To construct databases of the classification criteria, we employ different levels of lexical knowledge in extracting information from legal documents in Chinese, and build a case instance for each closed lawsuit. Experimental results indicate that case-based classification outperforms keyword-based classification, and that machine-generated cases may offer performance accuracy that is about 7% below that of human-provided cases. Hoping to boost inference efficiency of our classifiers, we also design methods that merge the machine-generated criteria. Empirical results show that our methods can maintain the classification quality within 20% of the quality achieved by human-provided cases, even when we aggressively reduce the number of previously machine-generated cases by about seventy percents.

Keywords

Case Classification, Case Clustering, k-Nearest Neighbor, Case-based Reasoning, Rule-based Reasoning, Chinese Legal Information System

1. INTRODUCTION

Researchers of Artificial Intelligence and Law have explored several knowledge representation formalisms for legal information, and the research work has yielded several useful applications, e.g., CATO [3], HYPO [4], LEX [18], and ON-LINE [36]. Among the knowledge representation formalisms, case-based reasoning (CBR) has attracted a lot of attention, and the literature has seen several sophisticated techniques for CBR [25]. We have also seen applications of artificial neural networks (e.g., [29, 33]), conceptual networks (e.g., [27]), and decision trees (e.g., [9]) to the field. Inspired partially by recent progresses in legal informatics, and partially by increasing availability of Chinese legal documents on the Internet [1], there have been burgeoning interests in Chinese legal information systems, e.g., [12, 13].

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
ICAIL '03, June 24-28, 2003, Edinburgh, Scotland, UK.
Copyright 2003 ACM 1-58113-000-0/00/0000…$5.00.

Despite the achievements accumulated from applying CBR to legal reasoning, Brüninghaus and Ashley argue that we must have ways to construct effective cases more efficiently to enhance applicability of CBR systems to real-world problems. To this end, they apply machine-learning methods to abstract entities in legal documents for automatic indexing of cases in SMILE [10]. Moens et al. employ an array of text analysis techniques for automatic extraction of cases in SALOMON [30]. Weber extracts information about case attributes using text-template mining algorithms in PRUDENTIA [39]. Pannu applies genetic algorithms to learning important features of cases in CBR in GAINC [31], and employs the "near miss" concept [41, pp. 234] based on the Euclidean distance in the learning processes.

In this paper, we report results of applying human-provided and machine-generated cases in a CBR-like system for classifying criminal summary judgments in Taiwan. Given a legal document, we sift its contents for potentially useful information, and use the results to build a corresponding case. After building cases for all relevant documents, we apply them for classification problems related to criminal summary judgments. Although the amount of these machine-generated cases remains manageable at the current stage, it is just a matter of time that we will have a large number of such raw cases. We must find ways to construct fewer but more representative cases from these raw cases so that we can conduct inference effectively and manage case databases efficiently. Hence, we also design a clustering algorithm that will convert a given case database into a more succinct counterpart that, we hope, would offer the same quality of support for lawsuit classification.

To evaluate performance of these machine-generated cases, we manually constructed rules and cases for our classification problems. Our system classifies lawsuits of unknown types into 12 categories, and selects applicable articles for lawsuits from 14 law articles. Experimental results indicate that human-provided cases outperform machine-generated ones by a margin of about 7% in precision and recall, when we apply the *k*-Nearest-Neighbor [2] principle in the decision making process. This performance difference will widen to about 20% when we gradually merge the case databases to 30% of their original sizes. Considering the economic costs of directly using human experts to construct and manage case databases for CBR systems, we find that computer software can assist this knowledge intensive task well.

Automatic construction of cases requires computer software to analyze the text information of legal documents, and is a language-dependent task. Chinese, like many other Asian languages, differs from English in many aspects [26]. For instance, words are not separated by spaces in both Chinese and Japanese texts. This and other differences, which we will explain shortly, post challenges to abstracting Chinese legal documents. We apply lexical analysis of the Chinese legal documents, and apply the results to lawsuit classification.

Section 2 provides more information about the challenges in processing Chinese text documents. Also included in the section are information about the criminal summary judgments in Taiwan and the source of our data that are used in constructing and evaluating the cases. Section 3 describes definitions of rules and cases in our system, and how we apply them to classify lawsuits. Section 4 elaborates on our algorithms for generation and clustering of cases.[†] Section 5 presents our experimental setups and results with analyses. Finally, we review our approach by comparing it with more related work, and discuss several possible extensions of our work.

2. BACKGROUND

2.1 Chinese Legal Document Processing

The following Chinese excerpt from a judgment document describes a drunk driver who caused a traffic accident. To protect the privacy of the involved individuals, we have replaced their given names with "Ｏ", and improvised plate numbers of their cars.

> 吳ＯＯ於民國九十年十月二十七日上午十時十分許，在某ＫＴＶ店內服用酒類，致其反應能力降低，已不能安全駕駛動力交通工具後，仍駕駛車號ＨＹ－１２３４號自用小客車沿板橋市文化路往台北方向行駛，在行經臺北縣板橋市文化路與站前路路口時，撞及自對向車道駛來，欲左轉站前路，由林ＯＯ所駕駛之Ｚ３－５６７８號自用小客車，嗣經警方處理，對吳ＯＯ施以酒精測試，其測定值為０．八五ＭＧ／Ｌ，始循線查知上情。

The processing of Chinese documents differs from that of English documents in several aspects [26]. Word separation is the most distinct difference among all others. While English words are separated by spaces, Chinese characters are separated only by punctuations. Hence, it is relatively harder for computer programs to determine word boundaries in Chinese [43]. Chinese is also more permissive in noun and verb generation. Native speakers of Chinese often use verbs that are not necessarily listed in the most complete dictionary, yet these speaker-invented verbs are still communicative in everyday conversations [35]. These so-called *unknown words* demand special treatments in high-quality Chinese understanding systems [14].

Despite the myriad of research work on Chinese information processing, there is relatively little work tailored for Chinese legal information. Lai and Huang use a small Chinese legal corpus in demonstrating the applicability of the Dependency Grammar for annotating Chinese documents [24]. Brown builds the CHINATAX system for inference problems that are related to the Commercial law of China [8].

[†] Because the word *case* can refer to both legal cases in everyday conversation and cases in case-based reasoning, we will use different words to refer to these two concepts when necessary. We use *lawsuits* for legal cases and *cases* for the data used in case-based reasoning. When the context provides sufficient clues for disambiguation, we continue to use *case* for both senses.

Abstracting Chinese legal documents for legal inference or law education requires mature information retrieval and extraction techniques for Chinese. For instance, to generalize from *entities* to *roles* like SMILE does [10], we need to identify Chinese names and verbs with reasonable accuracy in the first place, which, unfortunately, demands some more research work still.

For this study, we resort to strategies commonly used in Chinese information retrieval systems. We segment consecutive characters into words with the help of a dictionary. Specifically, we employ HowNet [20] for determining the word boundaries in Chinese legal texts. To compensate the fact that HowNet is not designed particularly for the legal domain, we also customize HowNet by adding legal terms to it in our experiments.

It is possible that there are multiple ways to segment a character string into words. When a character substring may be used to match more than one listed word, we prefer the longest word. For instance, assuming that "基於傷害之故意", "傷害" and "故意" are listed in the dictionary, there are two ways to split "基於傷害之故意". The first alternative is to leave the original string intact, and the second is to split it into several parts, i.e., "基", "於", "傷害", "之", and "故意". Since we prefer the longest match, we accept the first segmentation.

2.2 Criminal Summary Judgments in Taiwan

The public prosecutors of the government are responsible for instituting a prosecution against offenders of legally forbidden activities. Some of the criminal lawsuits are relatively less serious, and are judged in the court for criminal summary judgments. After the policemen collect supporting evidence, the public prosecutor sues the offender by issuing an indictment document and sending the defendant to the court. The judge then examines the evidence, conducts further investigation if necessary, and makes a judgment as to whether and how the defendant should be sentenced. To publicize the judgments, the judge composes a document that contains the sentence as well as findings of the investigation into the relevant evidence and arguments.

We select some classes of the criminal summary judgments in our experiments, and list these selected classes in Table 1 on the next page. The **Chinese** column shows the official names of the crimes, and the **English** column shows the translation of these official names in English. The **Symbol** column shows codes of the crimes that we will use as crime identifiers when we report the accuracy achieved by our classification and clustering algorithms.

2.3 Data Source

We acquired the documents of the closed cases from the Panchiao district court where one of the authors serves as a judge. We obtained two consecutive seasons of realistic documents. The majority of these real cases, but not all, belong to the offences listed in Table 1. We got documents for 503 lawsuits from each of these two seasons. We used the lawsuits occurred in the first season as the training data for computers to generate cases, and used the lawsuits occurred in the second season as the test data for evaluating quality of the rules and cases constructed by human and machine-learning algorithms. The collected lawsuits covered all 12 different types of crimes listed in Table 1, and reflected the frequencies of lawsuits in real life. Unfortunately, some crime types are far more often than others, and the distribution over the crime types, listed in Table 2, was not as even as one might like to have for evaluation of algorithms.

Chinese	English	Symbol
公共危險罪	Offences against Public Safety	C_1
妨害風化罪	Offences against Morals	C_2
賭博罪	Offences of Gambling	C_3
傷害罪	Offences of Causing Bodily Harm	C_4
竊盜罪	Offences of Larceny	C_5
侵占罪	Offences of Misappropriation	C_6
贓物罪	Offences of Receiving Stolen Property	C_7
違反動產擔保交易法	Offences against Chattel Secured Transactions Act	C_8
違反毒品危害防制條例	Offences against Statute for Narcotics Hazard Control	C_9
違反電子遊戲場業管理條例	Offences against Electronic Game Arcade Business Management Act	C_{10}
違反兒童與少年性交易防制條例	Offences against Child and Juvenile Sexual Transaction Prevention Act	C_{11}
違反台灣地區與大陸地區人民關係條例	Offences against Statute Governing the Relations between the People of the Taiwan Area and the Mainland Area	C_{12}

Table 1. Types of offences in Chinese, English, and symbols

Symbol	C_1	C_2	C_3	C_4	C_5	C_6	C_7	C_8	C_9	C_{10}	C_{11}	C_{12}	C_{13}
Season 1	158	26	30	14	99	15	9	15	19	16	19	9	74
Season 2	142	19	39	15	139	24	11	21	23	7	25	11	27

Table 2. Statistics of case quantities, where C_{13} denotes all other cases not belonging to the 12 selected categories

Each document in our collection contains several sections. The indictment document and the judgment document have similar structures. Both contain a header section that describes the information about the court and the prosecution/judgment date, a section that describes the information about the defendants, a section that describes the alleged/confirmed criminal activities, and a section that describes the alleged/judged offended law articles and sentences. As we explain in the next section, the classification component of our system takes indictment documents as input, and extracts the alleged-activity sections for classifying the lawsuits and selecting the offended law articles. In Section 4, we explain how the learning component of our system takes the judgment documents as input for automatic generation of cases.

3. HUMAN-MANAGED CLASSIFIERS

Since case classification is an ordinary job in the courts, we may ask human experts to specify rules and cases for case classification. In this section, we explain how this can be done for legal decisions using Chinese documents. In Section 5, we will compare the performances of these human-provided cases and machine-generated cases.

3.1 Keyword-based Rules

3.1.1 Rule Syntax and Semantics

We apply rules for determining prosecution reasons and selecting applicable sentencing articles. Figure 1 shows the syntax of our

```
1. prosecution reason
2. activation threshold
3. no
4. taboo-phrase-1
5. taboo-phrase-2
    ......
6. taboo-phrase-n
7. event
8. indicative-phrases-1: weight-1
9. indicative-phrases-2: weight-2
    ......
10. indicative-phrases-m: weight-m
```

Figure 1. Syntax for rules and cases

rules for judging prosecution reasons. The line numbers are given for explaining the semantics, but they are not part of the rules.

The syntax comprises of four parts. The first part, which occupies only the first line, contains the prosecution reason that will be assigned to the current lawsuit if the rule is fired. The second part, which occupies only the second line, contains the threshold that is used to determine whether the rule should be fired. The third part, which spans between lines 4 and 6, specifies the phrases that will prohibit the rule from being fired if any of them appears in the indictment documents. Line 3 is included here to indicate the beginning of the list of taboo phrases. The third part is optional, and not every rule requires a taboo list. The fourth part, which spans between lines 8 to 10, specifies the phrases that may activate the rule. Each indicative phrase has an associated weight that encodes its own strength for activating the rule, and phrases listed in the same line have the same weights. Similar to line 3, line 7 signifies the beginning of the list of activating phrases.

A sample rule for judging the prosecution reasons of cases follows. This is a rule for judging whether the case is an *offenses of causing bodily harm* (傷害) case. The rule will be fired if the total weight of indicative phrases found in the indictment document exceeds 20.

```
1.  傷害
2.  20
3.  event
4.  爭執、口角：10
5.  基於傷害之故意、基於傷害人身體之故意：10
6.  毆：10
7.  挫傷、傷害、擦傷、裂傷、死亡：10
```

The syntax of our rules for selecting sentencing articles is similar. The only difference is that we use the first line to store information about the applicable sentencing articles for the current lawsuit. We still need to check contents of the indictment document to select which article is applicable because different sentencing articles may apply to different situations under the same prosecution reason. Following is a rule for judging whether the 277th article of the criminal law (刑法) should be applied.

```
1.  刑法第 277 條
2.  10
3.  event
4.  基於傷害之故意、基於傷害人身體之故意：10
```

3.1.2 Rule Applications

Given a rule, the task of case classification boils down to string matching, given that we are not doing in-depth semantic analysis of Chinese documents yet. We examine whether the required and the forbidden strings appear in the **s**ection for **a**lleged **c**riminal **a**ctivities (**SACA**) in the indictment documents to determine the category of the given case.

Recall that Chinese is flexible in noun and verb generation. It is possible that public prosecutors use slightly different phrases to describe the same concept. For instance, one may use "危害他人生命" and "危害他人性命" to describe the concept "endanger other's life." Although these character strings are different, they actually represent the same meaning. We cope with this problem by comparing strings with the longest common substring (LCS) algorithm [19]. Given the previous two Chinese strings, the LCS algorithm is able to find their longest common substring "危害他人命", and returns 5 as its output. Since LCS employs the concept of dynamic programming, we can efficiently conduct string matching for selecting rule to apply. If the matched substring represents a sufficient proportion of a listed string in the rule, we will consider the listed string is found in the SACA. We set a threshold for deciding whether the substring can be consider as a match, and this threshold is set to 75% in our experiments.

Once we complete the string-matching step, we may fire the rule that receives the highest score, when we apply the Nearest-Neighbor principle. We will explain how we employ the *k*-Nearest-Neighbor principle later. Notice that if any forbidden string of a rule appears in the SACA, the rule will not be fired no matter what. For all other candidate rules, we accumulate the scores of the matched listed strings. During the accumulation, we allow each listed string only one chance to contribute to the scores so that duplicated strings will not bias the final decisions. Notice that, although the syntax for our rules allows different weights for terms, we assign the same weights to all terms in our current implementation.

3.2 Case-based Method

The syntax of our cases is the same as that of our rules. Their semantics are also similar, except that the ordering of strings in the fourth part of case specification must be observed, and that the weights for terms may be different. The ordering represents the presumed sequence of events, so is very important for decision-making using cases.

As a result of this required ordering of events, we cannot assign scores for all matched strings. The relative locations where the strings appear in the SACA determine whether the word will count. In the previous rule for "傷害" , a string that matches "挫傷" must appears after another string that matches "爭執" for the rule to get 10 points.

4. INSTANCE-BASED LEARNING

Building rule and case databases by human experts may provide better classification quality, but the time costs may become unacceptable when we confront real-world problems. In this section, we let computer programs generate and merge rule and case instances by extracting relevant information from legal documents. We use these machine-generated instances to classify the current lawsuit, and we refer to these machine-generated instances not as cases because they look too primitive to be called cases.

4.1 Instance Generation

As discussed in Section 2.3, each judgment document contains a section for the confirmed criminal activities and a section for the judged offended law articles. Therefore, we can extract these sections from each given document to compose an instance. Since the judged offended law article carries explicit information about the prosecution reason and the offended law article, we can save the information in the instances. We describe details about how we utilize the extracted sections next.

There are three alternatives for us to generate instances, and the main differences are whether we use a dictionary to segment the character strings into words and whether the results are used as cases or rules.

Before going into the instance generation step, we remove part of the given text during the preprocessing step. Treating each character string that is separated by punctuations as a unit, we remove units that contain two special sets of characters. The first set is {年, 月, 日, 時, 分}, and the second set is {市, 縣, 路, 村, 里, 段, 巷, 弄, 號}. The first set carries information about time, and the second carries information about locations. Cases of summary criminal judgments are relatively simple lawsuits, so the confirmed criminal activities typically occur during a particular time interval. Hence, the time stamps of the activities are not very important for summary judgments. We remove the location information based on the same reason.

After the preprocessing, we create an instance by removing commas and periods from the given text. Given the Chinese text that appeared in Section 2.1, we create the following instance. We cannot explain the meaning of the following Chinese text clearly here, but the passage still contains the description for the main criminal activities, i.e., driving under the influence of alcohol.

在某ＫＴＶ店內服用酒類致其反應能力降低已不能安全駕駛動力交通工具後撞及自對向車道駛來欲左轉站前路由林ＯＯ所駕駛之Ｚ３－５６７８號自用小客車嗣經警方處理對吳ＯＯ施以酒精測試其測定值為０・八五ＭＧ／Ｌ始循線查知上情

The second method for generating instances is a bit more complex. After the preprocessing, we use a dictionary to segment the character strings, and keep only words with more than two characters that are listed in the dictionary. Given the Chinese text that appeared in Section 2.1, we create the following word list.

內服 反應 能力 降低 不能 安全駕駛 動力 交通工具 車道 左轉 駕駛 自用 客車 警方 處理 施以 酒精 測試 測定

The word "安全駕駛" is not listed in the original HowNet dictionary, but is included as a word in our domain-dependent lexicon, so is identified as one word in our system. We interpret this list of keywords as an *ordered* list, and use it as a case instance. Although this instance is vaguer than the previous, it still contains the keywords that describe the criminal activities in the original indictment document.

The last method is similar to the previous method. We adopt all steps that lead to the generation of the word list. The only difference is that we interpret the word list as an *unordered* list when we apply these instances for classification. We do not need to store repeated words when we compute the word list, if we would apply instances of this third type only for classification. However, since we will merge instances as we discuss in Section 4.3, we must keep the repeated words in instances for collecting statistics of keyword frequencies. Notice that instances of this type are just like the rules that we present in Section 3.1, when we look for keywords for classification without concerning order and relative importance of the keywords.

The instance generation step need not create weights for the terms in our current implementation. We apply these machine-generated instances with the typical instance-based learning methods [2] that we explain in the next subsection.

4.2 Instance Applications

Given a set of instances generated by methods discussed in Section 4.1, we can apply the *k*-Nearest-Neighbor principle for classification. To this end, we must define a measure for "similarity" so that we can establish the notion of nearest neighbors of the current lawsuit. We define different similarity measures for the three types of machine-generated instances.

Instances of the first type are simply concatenated character strings. When we apply these instances for classification, we extract the SACA from the indictment document, and convert it into a long concatenated string using exactly the same method for generating instances of the first type. This long string, *X,* is compared with each stored instance, *Y*, by the LCS algorithm to obtain the length of their longest common sequence. The similarity between the current lawsuit and the prior case being compared is defined by s_1, where *Length* is a function that returns the number of characters in its input string.

$$s_1(X,Y)=\frac{1}{2}\left(\frac{LCS(X,Y)}{Length(X)}+\frac{LCS(X,Y)}{Length(Y)}\right)$$

LCS(X,Y)/Length(X) represents the proportion of the longest common sequence in *X*. *LCS(X,Y)/Length(Y)* carries an analogous interpretation based on the length of *Y*. Since the lengths of *X* and *Y* may be different, it would be arbitrary if we choose only one of these two ratios as the similarity measure. Hence, we take their average as the similarity measure. Using averages in the following similarity measures, s_2 *and* s_3, is based on the same reason.

Instances of the second type are lists of ordered keywords. Hence, we convert the SACA of the current lawsuit into an ordered list using the same method for generating instances of the second type. This ordered list, *X*, is then compared with each prior case, *Y*. Taking the order of keywords into consideration, the comparison will find the number of common keywords that appear in the same order in both *X* and *Y*. Let *OCW* be a function that will return the number of <u>o</u>rdered, <u>c</u>ommon <u>w</u>ords in its input word lists, and *Counts* be a function that will return the number of words in its input word list. The similarity between the current lawsuit and a prior case of the second type is defined as follows.

$$s_2(X,Y)=\frac{1}{2}\left(\frac{OCW(X,Y)}{Counts(X)}+\frac{OCW(X,Y)}{Counts(Y)}\right)$$

Because instances of the third type are similar to those of the second type, similarity measures for them are also very similar. Let *UCW* be a function that will return the number of <u>u</u>nordered <u>c</u>ommon <u>w</u>ords in its input word lists. When the SACA of the current lawsuit, *X*, contains repeated words that also appear in the prior case, *Y*, the repeated words will be counted as many times as they appear in the SACA. In contrast, repeated words in *Y* are counted only once. The similarity measure for the third type of instances follows.

$$s_3(X,Y)=\frac{1}{2}\left(\frac{UCW(X,Y)}{Counts(X)}+\frac{UCW(X,Y)}{Counts(Y)}\right) \qquad (1)$$

4.3 Instance Clustering

One problem of our approach to generating a case for each prior case is the monotonically growing size of the case database. If we keep adding new cases into the case database, we have to wait a long time for classifying a new case by spending time on case comparison. One way to alleviate this problem is to index the cases with smarter ways for efficient case retrieval. For instance, Moens et al. index legal cases with the help of text processing techniques [30].

An alternative is to identify similar case instances, and somehow merge them into one case. Clearly not all our machine-generated cases are required, and, if we can identify and keep those that are more representative than others, we may achieve the same accuracy of classification at reduced computational costs. The idea is similar to data clustering [22], where we cluster raw data into meaningful patterns of reduced space complexity to gain insight into the application of interests, and the patterns remain as effective as the original raw data for the target applications.

To this end, we must define a measure for "similarity" for merging instances. Since our original instances are already tagged with the classes, i.e., the prosecution reasons and offended articles, we can just merge instances with the same tagged classes. Let *X* and *Y* be two such machine-generated instances of the first or the second type. We define the size, *Size(X)*, of an instance *X* as follows.

$$Size(X)=\begin{cases}Length(X) & \textit{if } X \textit{ is of the first type}\\ Counts(X) & \textit{if } X \textit{ is of the second type}\end{cases}$$

Similarly, depending on the types of *X* and *Y*, we define the overlapping portion, *Com(X,Y),* of *X* and *Y*. If they belong to the first type, *Com(X,Y)* is set to the longest common substring of *X* and *Y*, and, if they belong to the second type, *Com(X,Y)* is set to the ordered common keywords in *X* and *Y*. As a result, depending on the types of *X* and *Y*, *Size(Com(X,Y))* must be equal to either *LCS(X,Y)* or *OCW(X,Y)*.

```
Procedure Merge2Instances(X,Y)
if ((Size(Com(X,Y)) ≥ p*Size(X)) and
    (Size(Com(X,Y)) ≥ p*Size(Y))) {
  Remove X and Y from the instance database;
  Add Com(X,Y) into the instance database; }
else if ((Size(Com(X,Y)) < p*Size(X)) and
        (Size(Com(X,Y)) ≥ p*Size(Y)))
  Remove Y from the instance database;
else if ((Size(Com(X,Y)) ≥ p*Size(X)) and
        (Size(Com(X,Y)) < p*Size(Y)))
  Remove X from the instance database;
```

Using these definitions, we merge two instances using the following `Merge2Instances` procedure, where *p* is a percentage threshold selected by the instance database manager. When *X* and

Y are really similar to each other, their common part will constitute a significant portion of both. In this case, we replace both cases with their common portion. If the common portion exceeds a significant portion of just one of original cases, say *Y*, we can remove *Y* and keep *X* that still carries the common information.

To merge all instances in the same class, we temporarily assign an identification number to each case. Let `IDB` be the instance database that contains the instances being considered to merge, `I[j]` be the `j`th instance in the `IDB`, and `S` be current number of instance in the `IDB`. We use the following simple procedure to merge all instances in the `IDB`. Notice that `S` may decrease as we remove instances from the `IDB` when we call *`Merge2Instances`*.

```
Procedure MergeAllInstances(IDB)
S = current size of the IDB
do {
    for j = 1 to S-1
        for k = j+1 to S
            Merge2Instances(I[j], I[k]);
} while (new cases were added to the IDB);
```

There are two alternatives for merging instances of the third type. We can collect statistics about the keyword frequencies in the learned instances, and keep those keywords that are neither too frequent nor too infrequent. Let *n* be number of instances being considered to merge. As we just discussed, we assume that these *n* instances must have been tagged with the same class value. Also assume that each distinct keyword contained in these *n* instances is assigned an ID number, and that *k*[*m*] be the total number of times the *m*th distinct keyword appears in these *n* instances. We merge these *n* instances into just one instance that contains such keywords that meet the following criterion.

$$0.5 \le k[m]/n \le 1.5$$

For crime type classification, since our current study covers only those 12 crime types listed in Table 1, we will have one standard instance for each of the crime type. Notice that this is almost like that, for each crime type, we mechanically generate one rule using the syntax and semantics defined in Section 3.1, except that we do not generate the rule threshold and the term weights.

The preceding strategy may seem to be very extreme. The other alternative is to take back some of the removed instances with the help of the 12 typical instances. We recover the instances whose keyword list cover 50% or less of the keyword list of the typical instance of the same categorization. Notice that, in this case, duplicated words in the recovered instances will receive multiple counts when we compute *Counts(Y)* in (1).

5. EMPIRICAL COMPARISONS

In the preceding sections, we discuss several possible decision factors for constructing a CBR-like system for lawsuit classification in a Chinese environment. We summarize these factors before we present experimental results that reflect effects of these factors, and show the structure of our experiments in Figure 2. The first factor is whether we use rules or cases, and the second is whether we build the rules and cases by human experts or algorithms.[‡] If we algorithmically generate cases, we must also choose a way to deal with the word segmentation problems in Chinese, and we have discussed three possible alternatives in Section 4.1. Two of these alternatives lead to the generation of cases, and the last to the generation of keyword-based rules. After we choose the form of Chinese information in the cases, we have to consider how to merge the cases. Specifically we must determine the percentage threshold, *p*, in *`Merge2Instances`* for merging cases. On the other hand, we have picked two particular ways to merge the keyword-based rules in Section 4.3. Once we have a case database or a rule base, we may apply the *k*-Nearest-Neighbor (*k*NN) or simply the Nearest-Neighbor (NN) principle for classification, but we do not explicitly show this factor in Figure 2.

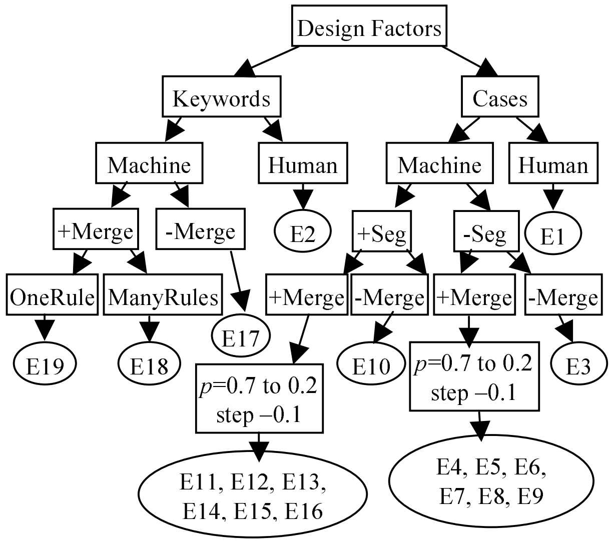

Figure 2. Structure of our experiments

In Figure 2, the rectangles show the distinguishing features of the experiments. The ovals show the identification codes of the experiments when we discuss the experimental results. In experiments E4, E5, E6, E7, E8, and E9, we set *p* to 0.7, 0.6, 0.5, 0.4, 0.3, and 0.2, respectively. Similarly, we set *p* to 0.7, 0.6, 0.5, 0.4, 0.3, and 0.2 in experiments E11, E12, E13, E14, E15, and E16, respectively.

We evaluated effects of these decision factors by testing different system designs by the same set of test documents that we gathered in the second season, while we used data gathered in the first season to create the case database. In the experiments, we let the classifiers choose to put the test lawsuit into the "unknown" class, which represented the fact that the classifiers knew that the test lawsuit did not belong to any offenses listed in Table 1.

5.1 Performance Measures

The evaluation was based on the standard definitions for precision, *P*, and recall, *R*. However, for our classification problems, precisions were considered to be more important than recalls. If the classifiers put a lawsuit in the "unknown" class, human experts could classify these unclassified cases. Should the classifiers misclassify a lawsuit, more human labor would be required to detect

[‡] From now on, we refer to the machine-generated rule and case instances as rules and cases for simplicity of writing.

and correct the errors. Therefore, we also measured systems' performances with the *F* measure. The *F* measure, shown below [28], is a nonlinear combination of precision and recall, and weights them differently by manipulating the parameter β. When β is 1, *P* and *R* are weighted equally; when β approaches infinity and zero, *F* approaches *R* and *P*, respectively. In the following reports, we set β to 0.5.

$$F = \frac{(\beta^2+1)PR}{\beta^2 P + R} \tag{2}$$

The following table shows the precisions and recalls of our system when we used cases and rules that were provided by human. The left portion of the **EXPID** column indicates whether the statistics are for cases or rules by the codes used in Figure 2, and the right portion indicates whether the statistics are precisions (P), recalls (R), or *F* measures (F). We do not show the percentage signs for the data in Table 3 because of the limited table width.

EXPID	C_1	C_2	C_3	C_4	C_5	C_6	C_7	C_8	C_9	C_{10}	C_{11}	C_{12}	C_{13}
E1-P	100	100	95	57	99	82	100	100	100	100	100	100	41
E2-P	98	82	96	62	90	63	100	95	100	35	100	100	28
E1-R	93	89	97	80	99	75	36	86	96	71	100	100	70
E2-R	95	95	59	53	100	21	27	90	87	100	80	100	52
E1-F	99	98	95	61	99	80	74	97	99	93	100	100	45
E2-F	97	84	85	60	92	45	65	94	97	40	95	100	31

Table 3. Detailed performance statistics for rules and cases

When comparing two approaches for constructing classifiers, we found that a classifier seldom performed better than the other for all crime types, although one may outperform the other most of the time. Data in Table 3 support this observation. Comparisons between E1-P and E2-P, E1-R and E2-R, and E1-F and E2-F suggest that cases are better than rules for classifying most, but not all, crime types. In addition, some crime types are intrinsically easier than others to be classified correctly, while others are relatively harder. Using human-provided cases, we could hardly surpass 80% in all measures for C_4. Due to this reason and the page limitation for this paper, we will not examine the precision, recall, and F measures for individual crime types henceforth. Instead, we analyze their averages. Let p_i be the precision for the i^{th} crime type. We report the direct average precision *AP* and the weighted average precision *WP*. When computing *WP*, we weight individual precision p_i by the number of cases, c_i, that belong to the i^{th} crime type in the test data.

$$AP = \tfrac{1}{13}\sum_{i=1}^{13} p_i \tag{3}$$

$$WP = \sum_{i=1}^{13} c_i p_i \Big/ \sum_{i=1}^{13} c_i \tag{4}$$

The averages, *AR* and *WR*, for recalls are defined analogously. To compute the *AF* and *WF*, we first compute the *F* measures of all crime types by (2), and apply formula analogous to (3) and (4) in the calculations. Consequently, although individual f_i falls between corresponding p_i and r_i, *AF* does not have to fall between *AP* and *AR*, neither does *WF* have to fall between *WP* and *WR*.

Using the direct averages is tentative to say that all crime types are equally important. The relative frequencies of occurrences of each crime type do not matter. In contrast, using the weighted averages presumes that all individual lawsuits are equally important.

5.2 NN vs. *k*NN

Our *k*NN method was a variant of the weighted nearest neighbor methods [15], and Thompson applied another variant of *k*NN for categorization of Case law [34]. We did not set a static *k* when we used *k*NN for classification. Instead, we set a static threshold for dynamically determining this *k*. Only prior cases whose similarity measures were 0.3 or larger could cast votes on the class of the current lawsuit. The current lawsuit was assigned to the class that received the most votes. If there were more than two classes that received the same number of votes, we computed the total of the similarity scores of all voting cases in the competing classes. The class that had the largest total score was assigned to the current lawsuit.

Classifiers that used our *k*NN principle dominantly performed better than their counterparts that used the NN principle, excluding very rare exceptions. Hence, we report performance statistics of our classifiers that used our *k*NN principle in the remainder of this section.

5.3 Classifying Prosecution Reasons

The following table shows the statistics for the experiments that are shown in Figure 2. The SIZE column shows the number of cases or rules used in the corresponding classifiers. The parenthesized numbers in the EXPID column show the values of the parameter *p* used in `Merge2Instances` in the experiments.

We computed the weighted average precision, recall, and F measures with the case amounts shown in Table 2, and these weighted measures make the performance of all classifiers look better than the direct average ones would. This is because, most of the time, the classifiers achieved higher performance in crime types that include relatively more lawsuits for instance learning and testing.

EXPID	SIZE	*AP*	*AR*	*AF*	*WP*	*WR*	*WF*
E1	24	90%	84%	88%	94%	91%	93%
E2	12	81%	74%	76%	88%	84%	86%
E3	429	86%	80%	81%	92%	87%	90%
E4 (0.7)	321	86%	79%	81%	92%	87%	90%
E5 (0.6)	262	84%	74%	78%	92%	80%	88%
E6 (0.5)	201	83%	63%	69%	90%	74%	83%
E7 (0.4)	130	80%	59%	67%	89%	53%	74%
E8 (0.3)	64	59%	39%	42%	82%	27%	42%
E9 (0.2)	20	53%	44%	51%	49%	45%	48%
E10	429	86%	77%	80%	91%	87%	88%
E11 (0.7)	283	86%	76%	80%	91%	87%	88%
E12 (0.6)	215	83%	76%	79%	90%	85%	88%
E13 (0.5)	136	79%	72%	75%	90%	82%	87%
E14 (0.4)	92	75%	59%	67%	88%	75%	83%
E15 (0.3)	47	72%	53%	60%	84%	54%	71%
E16 (0.2)	18	60%	43%	55%	79%	34%	62%
E17	429	74%	62%	65%	76%	80%	73%
E18	338	62%	61%	61%	72%	80%	72%
E19	12	57%	54%	50%	75%	51%	51%

Table 4. Experimental results for lawsuit classification

Overall, classifiers that used cases performed better than classifiers that used keyword-based rules. This is supported by the statistics of E1 and E2, where human experts provided the cases and rules. Also, despite the detailed differences between experiments from E3 through E16 and experiments from E17 through E19, the former group that used machine-generated cases achieved better performance than the latter group that used machine-generate rules. The statistics for E9, E16, and E19 suggest that only when we aggressively merged machine-generated cases might the machine-generated keyword-based rules offered better performances.

Classifiers that used human-provided information outperformed classifiers that used machine-generated instances. Statistics for E1 are better than those for E3 through E16, and statistics for E2 are better than those for E17 through E19. As we noted previously that statistics in Table 4 are for *k*NN versions of the classifiers. Had we shown the statistics for the NN versions, the performance differences between the classifiers that used human-provided and machine-generated instances will widen by more than 5%.

Next, we look into the effects of conducting word segmentation in the Chinese documents by comparing the data for the pairs, E3 and E10, E4 and E11, E5 and E12, E6 and E13, E7 and E14, E8 and E15, and E9 and E16. Segmenting Chinese characters into words led to more compact case databases after we merged cases. When we merged the cases relatively conservatively ($p \geq 0.6$), cases that used segmented words offer slightly inferior performance. In contrast, when we merged cases more aggressively ($p \leq 0.5$), cases that used segmented words offered more robust performances.

Merging the cases does not always reduce the classification accuracy, due to similar prior cases in our training data. The performances of E3, E10, and E17 were similar to those of E4, E11, and E18, respectively, although we cut the sizes of the databases by about a quarter. As we merged cases more and more aggressively, the performance declined almost monotonically from E3 through E9 (and from E10 through E16). Performances of the classifiers that used non-segmented Chinese characters deteriorated faster than those classifiers that used segmented Chinese words.

Finally, it is interesting to examine the performances of our classifiers when we merged the case database such that the database contained approximately the same number of cases as human-provided case database did. This difference sheds light on the range of difference between human experts and a fully automated case generation mechanism. The statistics show that the average differences are 34% (E1-E16) and 42% (E1-E9) for case generation and only 13% for rule generation (E2-E19).

5.4 Assigning Offended Articles

We applied similar methods and conducted similar experiments for assigning offended articles to lawsuits. We reused the data that we collected for lawsuit classification, but focused on their cited articles. Most of our training and test data used one of the fourteen articles. These fourteen articles are related to the twelve prosecution reasons that we used in the previous subsection. The number of applicable articles is larger than the number of prosecution reasons because different articles may be applied to different criminal activities even when they are accused under the same prosecution reason.

Assigning articles relies on more detailed knowledge of criminal activities, and therefore demands more thorough analysis of the indictment documents. For instance, the Chinese excerpt shown in Section 2.1 describes the blood alcohol level of the drunk driver, and this information is very important in selecting the offended law articles because, in Taiwan, the penalty against drunk driving depends on the blood alcohol levels of the offenders. At this stage of our work, we have not applied text analysis techniques to extract this particular information yet. Problems of this kind contribute a lot to the inferiority in performance of our article assignment subsystem.

We encode these 14 groups of articles with A_1, A_2, …, and A_{14}, and all other articles with A_{15}. Table 5 shows the amount of lawsuits that we collected from the first (S1) and the second (S2) season.

	A_1	A_2	A_3	A_4	A_5	A_6	A_7	A_8	A_9	A_{10}	A_{11}	A_{12}	A_{13}	A_{14}	A_{15}
S1	158	17	20	7	9	91	8	12	9	15	19	16	19	9	94
S2	136	19	29	6	10	127	12	20	11	21	23	7	25	11	46

Table 5. Number of lawsuits that use the designated articles

We used the data collected from the first and the second season as the training and test data, respectively. Table 6 shows the experimental results, where E20, E21, …, and E38 respectively take the positions of E1, E2, …, and E19 in Figure 2.

EXPID	SIZE	*AP*	*AR*	*AF*	*WP*	*WR*	*WF*
E20	22	87%	84%	84%	92%	89%	91%
E21	17	78%	73%	74%	88%	80%	85%
E22	409	67%	66%	65%	81%	83%	81%
E23 (0.7)	308	79%	69%	74%	85%	83%	83%
E24 (0.6)	243	75%	65%	71%	83%	77%	81%
E25 (0.5)	201	79%	64%	70%	85%	77%	81%
E26 (0.4)	117	79%	52%	65%	85%	54%	71%
E27 (0.3)	52	60%	36%	43%	79%	31%	42%
E28 (0.2)	19	40%	23%	24%	50%	20%	20%
E29	409	79%	68%	72%	82%	83%	81%
E30 (0.7)	283	81%	72%	75%	84%	84%	83%
E31 (0.6)	194	85%	71%	75%	86%	83%	83%
E32 (0.5)	125	84%	67%	76%	88%	81%	85%
E33 (0.4)	87	84%	69%	75%	86%	76%	81%
E34 (0.3)	43	77%	54%	64%	83%	59%	73%
E35 (0.2)	17	58%	43%	48%	75%	38%	54%
E36	409	63%	58%	62%	69%	77%	67%
E37	315	50%	54%	50%	64%	74%	65%
E38	14	50%	48%	48%	64%	73%	65%

Table 6. Experimental results for article assignment

The experimental results reflect the fact that assigning articles is more difficult than classifying lawsuits from two aspects. We can verify that the average quality of lawsuit classification is better than that of article assignment. We can also see that the differences between the classification quality of the machine-generated ($p \geq 0.6$) and human-provided cases have widened.

Statistics in Table 6 support most of the observations that we found from statistics in Table 4. Cases offered better performance than keyword-based rules. Segmenting Chinese characters led to more compact case databases that offered comparable classification quality. We could merge cases modestly ($p \geq 0.6$), and maintained assignment quality at the same time.

6. DISCUSSIONS

We report experimental results and analyses for applying machine learning techniques to create a CBR-like system for lawsuit classification problems. Unlike GANIC [31] that attempts to find the best combination of factors from a bag of candidates by genetic algorithms, our work attempts to identify distinguishing words directly from the raw documents before merging the learned rules and cases. The sizes of the training and test data and the coverage of prosecution reasons and applicable articles in the current experiments are quite limited and can be expanded in the future.

The precision and recall measures achieved by the current system are relatively better than those achieved by Thompson's [34]. We believe that this is partially due to that test lawsuits in our experiments are relatively simple. At this moment, we consider lawsuits that have only one defendant who committed only one crime. Due to this constraint, we allow ourselves to choose only one prosecution reason and one applicable article in the experiments. In reality, the prosecution reasons are not perfectly mutually exclusively applicable to a criminal activity. There is not always a clean cut between the applicability of C_3 and C_{10} in Table 1, for instance. When ambiguities do occur, our classifiers may classify the lawsuits into a category that does not agree with the judge's decision. Expanding the current system to process cases that involve multiple defendants who cooperatively committed one or more crimes in some manners is under way.

Our counting duplicated words only once in *Counts(Y)* in (1) in E19 was an arbitrary design decision. Multiple occurrences of a keyword signal a very important clue to the judgment of the current lawsuit, so assigning higher weights to such keywords is clearly in demand. The problem is how much weight should be granted to these keywords so that their occurrences receive appropriate recognition while we do not let these keywords dominate the final judgment. This selection of weights is entangled with the consideration of the length of the CASA, and is a topic in our current study.

Researchers have noticed that computers are potentially helpful for drafting legislative, e.g., [37, 40], and judicial, e.g., [7], documents. We may apply our classifiers as a key component in a computer-assisted system for drafting or proofreading judgment documents. Experiences show that, in Taiwan, about 2% of judgment documents are not well written in that the cited criminal activities do not perfectly support the criminal law articles that are applied to the defendants. This problem can be alleviated if we draft or proofread the judgment documents by computers.

Researchers have also notice that a good drafting system requires much more than the capability of classifying the lawsuits. Utilizing structural knowledge about canonical legal documents may improve the quality of the drafts and facilitate automatic indexing of cases. Branting et al. show that ontology and discourse information make the drafting system more useful [7]. Bench-Capon and colleague have proposed that ontology and related techniques are useful for legal information systems in general [5, 6].

For Chinese legal document processing, we agree with these insightful findings. Instead of directly analyzing the original legal documents, one may attempt to annotate the documents with semantic information so that we can extract relevant information more precisely. Indeed, there are already some efforts toward this direction. For examples, Ebenhoch [16] employs the resource description framework, and Zarri [44] the semantic web for knowledge representation. Development in these directions can be very helpful for Chinese legal document processing.

Our generation and clustering of the cases depends just on the lexical information, and leaves much room for improvements. One possibility is to index cases with the help of summarization techniques like those used in SALOMON [30], and we are ready to apply our experience to this field [42]. Adding more reasoning capability for inferring the contextual information in the lawsuits will also help [17]. In addition, we believe that lexical information contained in thesauri can be helpful for processing Chinese legal documents [11, 38], although Brüninghaus and Ashley find techniques from information extraction more helpful [9, 10].

In addition to enhancing management of our case database, we would also like to apply Bayesian networks [23] to legal reasoning. Statistical and probabilistic inference techniques should help us for extending our current system to include computer-assisted sentencing [21, 32] in the drafting task.

ACKNOWLEDGMENTS

We owe a lot to the anonymous reviewers for their invaluable comments on both writing and technical problems in the manuscript of this paper. We have attempted to correct most of the writing problems in this paper, and plan to pursue research directions pointed out by the reviewers. This research was support in part by Grant NSC-91-2213-E-004-013 from the National Science Council of Taiwan.

REFERENCES

In the following listing, we use *AI* for all occurrences of *Artificial Intelligence* in the book titles.

[1] The following websites provide legal information for Taiwan and China as of May 20, 2003. (in Chinese)
The Judicial Yuan (http://wjirs.judicial.gov.tw/jirs/);
The Legislative Yuan (http://www.ly.gov.tw/);
The Ministry of Justice (http://www.moj.gov.tw/);
Lawbank Information Inc. (http://www.lawbank.com.tw/);
http://www.ordos.nm.cn/haoxia/navigation/zhengfa.htm.

[2] Aha, D., Kibler, D., and Albert, M. Instance-based learning algorithms, *Machine Learning*, 6, 37–66, 1991.

[3] Aleven, V. *Teaching Case-Based Argumentation Through a Model and Examples*, Ph.D. Dissertation, University of Pittsburgh, Pittsburgh, Ohio, USA, 1997.

[4] Ashley, K. D. *Modeling Legal Argument: Reasoning with Cases and Hypotheticals*, MIT Press, 1990.

[5] Bench-Capon, T. J. M. and Visser, P. R. S. Ontologies in legal information systems; the need for explicit specifications of domain conceptualizations, *Proc. of the 6th Int'l Conf. on AI and Law*, 132–141, 1997.

[6] Bench-Capon, T. J. M. and Visser, P. R. S. Open texture and ontologies in legal information systems, *Proc. of the 8th Int'l Workshop on Database and Expert Systems Applications*, 192–197, 1997.

[7] Branting, L. K, Lester, J., and Callaway, C. Automating judicial document drafting: A discourse-based approach. *AI & Law*, 6(2-4), 111–149, 1998.

[8] Brown, G. CHINATAX: Exploring isomorphism with Chinese law. *Proc. of the 4th Int'l Conf. on AI and Law*, 175–179, 1993.

[9] Brüninghaus, S. and Ashley, K. D. Toward adding knowledge to learning algorithms for indexing legal cases, *Proc. of the 7th Int'l Conf. on AI and Law*, 9–17, 1999.

[10] Brüninghaus, S. and Ashley, K. D. Improving the representation of legal case texts with information extraction methods, *Proc. of the 8th Int'l Conf. on AI and Law*, 42–51, 2001.

[11] Cammelli, A. and Socci, F. A thesaurus for improving information retrieval in an integrated legal expert system, *Proc. of the 9th Int'l Workshop on Database and Expert Systems Applications*, 619–624, 1998.

[12] Chang, C.-T., Ho, J.-H., and Liu, C.-L. Decision support for criminal summary judgment, *Proc. of the TAAI 7th Conf. on AI and Applications*, 178–183, 2002. (in Chinese)

[13] Chen, C.-S. Legal informatics in Germany, *China Law Journal*, 46(2), 79–85, 2001. (in Chinese)

[14] Chen, K.-J. and Bai, M.-H. Unknown word detection for Chinese by a corpus-based learning method, *Int'l J. of Computational linguistics and Chinese Language Processing*, 3(1), 27–44, 1998.

[15] Cost, S. and Salzberg, S. A weighted nearest neighbor algorithm for learning with symbolic features, *Machine Learning*, 10, 57–78, 1993.

[16] Ebenhoch, M. P. Legal knowledge representation using the resource description framework (RDF), *Proc. of the 12th Int'l Workshop on Database and Expert Systems Applications*, 369–373, 2001.

[17] Hafner, C. and Berman, D. The role of context in case-based legal reasoning: teleological, temporal and procedural, *AI & Law*, 10(1-3), 19–64, 2002.

[18] Haft, F., Jones, R. P., and Wetter. T. A natural language based legal expert system for consultation and tutoring—the LEX project, *Proc. of the 1st Int'l Conf. on AI and Law*, 75–83, 1987.

[19] Hirschberg, D. S. Algorithms for the longest common subsequence problem, *J. of the ACM*, 24(4), 664–675, 1997.

[20] HowNet. http://www.keenage.com

[21] Hutton, N., Patterson, A., and Tata, C. Decision support for sentencing in a common law jurisdiction, *Proc. of the 5th Int'l Conf. on AI and Law*, 89–95, 1995.

[22] Jain, A. K., Murty, M. N., and Flynn, P. J. Data clustering: a review, *ACM Computing Surveys*, 31(3), 264–323, 1999.

[23] Jensen, F. V. *Bayesian Networks and Decision Graphs*, Springer, 2001.

[24] Lai, T. B.Y. and Huang, C. Dependency-based syntactic analysis of Chinese and annotation of parsed corpus, *Proc. of the 38th Annual Meeting of the Association for Computational Linguistics*, 255–262, 2000.

[25] Leake, D. B. (ed.). *Case-Based Reasoning*, MIT Press, 1996.

[26] Lee, L.-S., Chien, L.-F., Lin, L. J., Huang, J., and Chen, K.-J. An efficient natural language processing system specially designed for the Chinese language, *Computational Linguistics*, 17(4), 347–373, 1991.

[27] Legrand, J. A contribution to indexing in legal information retrieval, *Proc. of the 8th Int'l Workshop on Database and Expert Systems Applications*, 213–218, 1997.

[28] Manning, C. D. and Schutze, H. *Foundations of Statistical Natural Language Processing*, The MIT Press, 1999.

[29] Merkl, D., Schweighofer, E., and Winiwarter, W. Exploratory analysis of concept and document spaces with connectionist networks, *AI & Law*, 7(2-3), 185–209, 1999.

[30] Moens, M.-F., Uyttendaele, and C. Dumortier, J. Abstracting of legal cases: The SALOMON experience, *Proc. of the 6th Int'l Conf. on AI and Law*, 114–122, 1997.

[31] Pannu, A. S. Using genetic algorithms to inductively reason with cases in the legal domain, *Proc. of the 5th Int'l Conf. on AI and Law*, 175–184, 1995.

[32] Schild, U. J. Intelligent computer systems for criminal sentencing, *Proc. of the 5th Int'l Conf. on AI and Law*, 229–238, 1995.

[33] Schweighofer, E. The revolution in legal information retrieval or: The empire strikes back, *The Journal of Information, Law and Technology*, 1999(1), 1999.

[34] Thompson, P. Automatic categorization of case law, *Proc. of the 8th Int'l Conf. on AI and Law*, 70–77, 2001.

[35] Tseng, H.-H., Liu, C.-L., Gao, Z.-M., and Chen, K.-J. A hybrid approach for automatic classification of Chinese unknown verbs, *Int'l J. of Computational Linguistics and Chinese Language Processing*, 7(1), 1–28, 2002. (in Chinese)

[36] Valente, A. *Legal Knowledge Engineering*, IOS Press, 1995.

[37] Voermans, W. and van Kralingen, R. Bringing IT support for legislative drafting one step further: From drafting support to design assistance, *Proc. of the 6th Int'l Conf. on AI and Law*, 259, 1997.

[38] Voorhees, E. Using WordNet for text retrieval, *WordNet: An Electronic Lexical Database*, MIT Press, 285–301, 1998.

[39] Weber, R. Intelligent jurisprudence research: a new concept, *Proc. of the 7th Int'l Conf. on AI and Law*, 164–172, 1999.

[40] Winkels, R. and den Haan, N. Automated legislative drafting: generating paraphrases of legislation, *Proc. of the 5th Int'l Conf. on AI and Law*, 112–118, 1995.

[41] Witten, I. H. and Frank E. *Data Mining*, Morgan Kaufmann, 2000.

[42] Wu, C.-W. and Liu, C.-L. Ontology-based text summarization for business news articles, *Proc. of the ISCA 18th Int'l Conf. on Computers and Their Applications*, 389–392, March 2003.

[43] Wu, Z. and Tseng, G. Chinese text segmentation for text retrieval: Achievements and problems, *J. of the American Society for Information Science*, 44(9), 532–544, 1993.

[44] Zarri, G. P. Semantic web and knowledge representation, *Proc. of the 13th Int'l Workshop on Database and Expert Systems Applications*, 63–67, 2002.